Часть II

ЗНАКОМЬТЕСЬ – МЕДИЦИНСКИЙ ЯЗЫК ДРАКОН

Глава 5

ПРЫЖОК ДРАКОНА: ИЗ КОСМОСА В БОЛЬНИЧНЫЙ КОРИДОР

КОСМИЧЕСКАЯ ОДИССЕЯ

В этой главе речь пойдет об истории создания медицинского языка ДРАКОН (рис. 18).

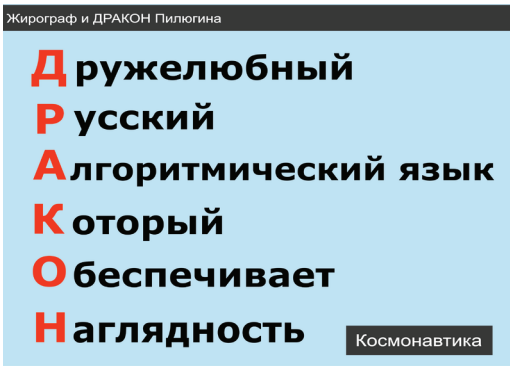


Рис. 18. Кадр из документального фильма «Жирограф и ДРАКОН Пилюгина» [225]

Судьба языка необычна. Он создавался вовсе не для медицины, а для космоса – в качестве языка программирования для орбитального корабля «Буран». При создании бортовых и наземных программ системы управления Бурана использовались языки Прол2, Диполь, Пси-Фортран, Лакс, Ассемблер (первые три разработаны в Институте прикладной математики имени М. В. Келдыша РАН). Обобщение опыта работы с этими языками привело к появлению ДРАКОНа [121, 122].

Цель разработчиков состояла в создании единого языка программирования и моделирования, который способен заменить специализированные языки: Прол2 (для разработки бортовых комплексных программ Бурана), Диполь (для создания наземных программ Бурана) и Лакс (для моделирования). Разработка языка ДРАКОН была завершена в 1996 году (спустя 3 года после неизбежного при крушении СССР, но грустного для участников закрытия программы Буран)[1].

За последние двадцать лет с помощью ДРАКОНа (и основанной на нем технологии разработки алгоритмов и программ ГРАФИТ-ФЛОКС) были созданы системы управления многих крупных космических проектов: «Морской старт», «Протон-М», «Фрегат», «Наземный старт», «ДМ-03»,

[1] Автор участвовал в разработке Бурана с первого до последнего дня. В то время я был начальником лаборатории комплексной разработки вычислительной системы Бурана.

«Ангара», южнокорейская ракета-носитель «KSLV» и др. [123–125].

Пуски ракет-носителей и космических разгонных блоков, при создании которых использовался и используется язык ДРАКОН, за истекший период выполнялись с шести космодромов мира, расположенных на трех континентах (Европа, Азия, Америка) и в океане:

1. космодром Плесецк;
2. космодром Байконур;
3. Европейский космический центр во Французской Гвиане «Kuru»;
4. южнокорейский космодром «Naro»;
5. международный плавучий космодром, производящий пуски с экватора в Тихом океане вблизи острова Рождества Республики Кирибати;
6. космодром Восточный.

Рис. 19. Старт комплекса «Энергия – Буран» 15 ноября 1988 года с космодрома Байконур

УДИВИТЕЛЬНОЕ И НЕОЖИДАННОЕ ПРОНИКНОВЕНИЕ В МЕДИЦИНУ

Альгирдас Каралюс (Литва) был первым, кто обратил внимание на возможность крупномасштабного использования языка ДРАКОН в медицине. Но не в качестве языка программирования, а совсем для других целей – в качестве графического средства для удобного описания последовательности действий врачей. То есть в качестве *медицинского алгоритмического языка*.

Инициативу Каралюса активно поддержали литовские врачи. За последнее время в Литве изданы четыре медицинских учебника на русском языке, в которых используется ДРАКОН [126].

1. Начальная неотложная акушерская помощь [127].
2. Специализированная реанимация новорожденного [128].
3. Неотложная медицинская помощь [129].
4. Травма [130].

Учебники апробированы в ряде стран (Литва, Казахстан, Азербайджан, Таджикистан, Туркменистан, Киргизия) в рамках курсов повышения квалификации врачей. Курсы проводили высококвалифицированные специалисты из Литовского университета медицинских наук (Lietuvos sveikatos

mokslų universitetas) для местных врачей. Проведенная апробация дала положительные результаты [131].

ГУМАНИТАРНЫЕ ТРЕБОВАНИЯ К ЯЗЫКУ ДРАКОН

Возникает вопрос: почему язык программирования ДРАКОН, созданный для орбитального корабля Буран и ракетно-космической отрасли, оказался удобным средством для описания медицинских алгоритмов?

Это легко объяснить. При разработке языка была поставлена амбициозная цель: ДРАКОН должен не только удовлетворять практическим нуждам космических систем, но и решать широкий круг задач, выходящих далеко за рамки ракетно-космической техники и программирования [132].

В связи с этим при создании языка ДРАКОН были выдвинуты необычные для программистов и математиков гуманитарные требования:

1. Улучшить работу человеческого ума.
2. Предложить эффективные средства для описания не только алгоритмов, но и структуры человеческой деятельности в любой отрасли знаний (включая бизнес-процессы).
3. Предоставить человеку такие языковые средства, которые резко упрощают восприятие сложных процедурных проблем и общение с коллегами, делают непонятное понятным. И за счет этого буквально заставляют человека мыслить отчетливо, глубоко и продуктивно. В этих условиях вероятность заблуждений, просчетов и ошибок неизбежно падает, а производительность растет.
4. Радикально облегчить межотраслевое и междисциплинарное общение между представителями разных организаций, ведомств, отделов, лабораторий, научных школ и профессий.
5. Устранить или уменьшить барьеры взаимного непонимания между работниками различных специальностей (врачами и физиками, биологами и математиками, конструкторами и экономистами и т. д.), а также программистами и теми, кто не владеет программированием [133, 134].

ЯЗЫК ДРАКОН. МЕДИЦИНСКИЙ ВАРИАНТ

Впоследствии была учтена медицинская специфика, чтобы профессиональные врачи могли легко и быстро описывать известные им, а также вновь создаваемые медицинские алгоритмы.

Для этой цели «космический» ДРАКОН был значительно сокращен и представлен в упрощенном виде, предназначенном специально для медиков и биологов. Были удалены все иконы (графические фигуры), свя-

занные с программированием. И оставлены только те, которые нужны для медицины.

В результате появился биомедицинский вариант языка, так называемый «Медицинский ДРАКОН».

УДОБНЫЕ ГРАФИЧЕСКИЕ ИНСТРУКЦИИ ДЛЯ ВРАЧЕЙ

Принято различать процедурные и декларативные знания (знания «как» и знания «что») [135]. Любые процедурные медицинские знания можно представить на медицинском языке ДРАКОН в виде графических инструкций для описания последовательности действий врачей. То есть в виде медицинских алгоритмов.

Примеры таких алгоритмов (инструкций) показаны в следующих главах.

ЧТО ДУМАЕТ ВРАЧ О МЕДИЦИНСКОМ ДРАКОНЕ

Травматолог-ортопед Василий Бачиашвили охарактеризовал медицинский ДРАКОН так:

> По моему мнению, ДРАКОН отлично подходит везде в медицине, где нужно передать процедурные знания. Это описания биологических процессов, алгоритмы диагностики и лечения и т. д.
>
> Было бы хорошо использовать ДРАКОН в методических пособиях и рекомендациях, в протоколах диагностики и лечения больных, в описании медицинских технологий, в медицинских книгах и журналах. Указанные материалы рассчитаны на то, чтобы большинство специалистов в данной области при чтении смогли их понять. Для того, чтобы в медицинской литературе начали использовать ДРАКОН, он должен стать максимально легким для освоения.
>
> Облегченный вариант ДРАКОНа (медицинский ДРАКОН) мне очень нравится. Думаю, это движение в правильном направлении» [218].

ВЫВОДЫ

1. Космическое происхождение медицинского языка сыграло важную роль, так как обеспечило тщательную отработку и надежность результатов, соответствующую высоким стандартам качества ракетно-космической отрасли.

2. В окончательном виде медицинский вариант языка ДРАКОН был сформирован в Литве по инициативе Альгирдаса Каралюса, литовских врачей и ученых, которые вложили огромный труд в его отработку и совершенствование.
3. Ученые и преподаватели Литовского университета медицинских наук (Lietuvos sveikatos mokslų universitetas) проанализировали «космический» ДРАКОН, упростили его, внесли полезные добавления и приспособили для медицинских нужд.
4. В Литве разработаны и опубликованы четыре медицинских учебника, в которых используется ДРАКОН.
    - Начальная неотложная акушерская помощь.
    - Специализированная реанимация новорожденного.
    - Неотложная медицинская помощь.
    - Травма.
5. Учебники апробированы в постсоветских странах: Литва, Казахстан, Азербайджан, Таджикистан, Туркменистан, Киргизия на курсах повышения квалификации врачей, которые проводили высококвалифицированные специалисты из Литовского университета медицинских наук.
6. Тщательные исследования языка ДРАКОН, проведенные литовскими врачами, и реальная практика его применения подтверждают целесообразность широкого использования языка для изображения медицинских алгоритмов в системе медицинского образования и в медицинской литературе (в медицинских учебниках, стандартах, руководствах, клинических рекомендациях, протоколах).

Глава 6

СПРАВОЧНИК: ГРАФИЧЕСКИЕ ФИГУРЫ ЯЗЫКА ДРАКОН

ЗАЧЕМ НУЖЕН СПРАВОЧНИК

При анализе дракон-алгоритмов иногда возникает необходимость вспомнить, как называется та или иная фигура. И зачем она нужна.

Для этого служит справочник фигур, представленный в этой главе.

ИКОНЫ МЕДИЦИНСКОГО ЯЗЫКА ДРАКОН

ДРАКОН – графический язык. Буквами этого языка являются геометрические фигуры, которые называются «иконы». Всего имеется 20 икон. Назначение икон показано на рис. 20. Кроме того, можно использовать выноски, как показано на рис. 3, 4.

Иконы должны быть заполнены текстом. Если текст отсутствует (икона пустая), значит это ошибка. Чаще всего используют иконы: *Действие* и *Вопрос*.

В иконе Действие пишут команду в повелительном наклонении, например:

- Обеспечь проходимость дыхательных путей.
- Выполни 30 компрессий грудной клетки.

В иконе Вопрос пишут да-нетный вопрос, т. е. вопрос, имеющий только два ответа: Да и Нет. Например:

- Есть ли реакция на прикосновение?
- Есть ли дефибриллятор?

МАКРОИКОНЫ МЕДИЦИНСКОГО ЯЗЫКА ДРАКОН

ДРАКОН имеет не только мелкие фигурки (иконы), но и крупные; они называются *макроиконы*.

Подобно тому, как слова состоят из букв, макроиконы (графические слова) слагаются из икон (графических букв). Медицинский ДРАКОН имеет 9 макроикон (рис. 21).

	Икона	Название иконы	Пояснение
1		Заголовок	В иконе «Заголовок» пишут название медицинского алгоритма, например, «Измерение кровяного давления»
2		Конец	В этой иконе пишут слово «Конец»
3		Действие	Указывают действие, которое должен выполнить медицинский работник или прибор
4		Вопрос	Да-нетный вопрос, т. е. вопрос, на который можно ответить либо Да, либо Нет. Все другие ответы запрещены
5		Выбор	Фраза (или вопрос), приглашающая выбрать один из вариантов
6		Вариант	Здесь пишут один из вариантов. (Число рассматриваемых вариантов равно числу икон «Вариант»)
7		Имя ветки	Эта икона обозначает начало ветки. В ней находится название ветки. (Ветка — это структурная часть алгоритма)
8		Адрес	Икона «Адрес» обозначает конец любой ветки, кроме последней. Она показывает, в какую следующую ветку надо идти
9		Вставка	Икона «Вставка» говорит, что в этом месте из медицинского алгоритма вынут «кусок», который перенесён в другое место. В иконе пишут название этого «куска»
10		Пауза	Икона «Пауза» задерживает выполнение действия. Время задержки пишут внутри иконы
11		Время	В иконе «Время» пишут длительность выполнения действия (или решения)
12		Время группы	Длительность выполнения группы действий. Не одного действия, а именно группы. Группа состоит из двух и более действий
13		Время группы *справа*	Икона «Время группы» присоединяется справа
14		Начало контрольного срока	В иконе пишут контрольное время критической процедуры. Например, «30 сек».
15		Конец контрольного срока	Указывают окончание контрольного времени. Например, «Прошло 30 сек».
16		Комментарий	Комментарий — это не действие. Это различные пояснения и подсказки, которые помогают быстрее понять алгоритм
17		Начало совместной работы	Означает НАЧАЛО одновременных скоординированных действий двух врачей
18		Конец совместной работы	Означает КОНЕЦ одновременных скоординированных действий двух врачей
19		Пояснение	Любые сведения, поясняющие икону, находящуюся слева от данной
20		Соединитель	Икона «Соединитель» используется при переходе с листа на лист (когда медицинский алгоритм размещается на нескольких листах)

Рис. 20. Иконы медицинского языка ДРАКОН

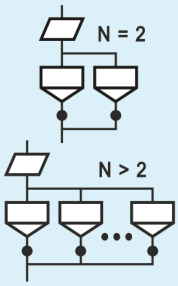
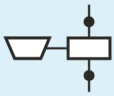
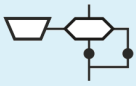
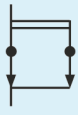
	Макроикона	Название макроиконы	Пояснение
1		Развилка	Черные точки — это валентные точки. В эти точки можно вставлять иконы, например, икону «Действие». Хотя бы одна валентная точка должна быть заполнена.
2		Переключатель (число вариантов 2 и больше)	Переключатель — это часть алгоритма, имеющая один вход вверху и один выход внизу. Внутри переключателя алгоритм разветвляется на несколько дорожек. Число дорожек равно двум и более. Переключатель строится с помощью иконы «Выбор» и нескольких икон «Вариант». Под каждой иконой «Вариант» имеется валентная точка.
3		Цикл	Цикл нужен для того, чтобы повторять действия. Повторение прекращается, когда будет выполнено условие. Условие записывают в иконе «Вопрос».
4		Веточный цикл	Веточный цикл нужен для того, чтобы повторять действия, расположенные в одной или нескольких ветках.
5		Действие и время	Справа нарисована икона «Действие». Слева к ней прицеплена икона «Время». Макроикона 5 показывает, что врач должен выполнить действие за указанное время.
6		Развилка и время	Справа нарисована «Развилка». Слева к ней прицеплена икона «Время». Макроикона 6 показывает, что врач должен принять решение за указанное время.
7		Время группы. *Время слева*	Справа нарисовано не одно действие, а группа, состоящая из двух (и более) действий. Слева к этой группе присоединена икона «Время». Макроикона 7 показывает, что время группы действий жестко задано.
8		Время группы. *Время справа*	То же самое, что в пункте 7. Отличие в том, что икона «Время» присоединена не слева, а справа.
9		Совместная работа	Означает синхронную, скоординированную работу врачей, включая: — начало совместных действий, — выполнение совместных действий, — конец совместных действий.

Рис. 21. Макроиконы медицинского языка ДРАКОН

Иконы и макроиконы – это строительные блоки, из которых сооружаются медицинские дракон-алгоритмы.

ВАЛЕНТНЫЕ ТОЧКИ

Важной частью макроикон служат *валентные точки* (на рис. 22 они показаны как черные кружки). В эти точки последовательно вводятся иконы и макроиконы, которые в совокупности образуют графический узор. После заполнения икон текстом узор превращаются в дракон-алгоритм.

МАРКЕРЫ МЕДИЦИНСКОГО ЯЗЫКА ДРАКОН

В сложном медицинском алгоритме могут появиться несколько веточных циклов. Как отличить их друг о друга?

Желательно иметь признак, позволяющий моментально, *по внешнему виду* обнаружить, где один цикл, а где другой. Примерно так же, как мы, не задумываясь, отличаем кошку от собаки и от вороны.

Отличительным признаком служит маркер. Предусмотрены четыре маркера, имеющие мнемонические названия: *сплошной, эллипс, вертикаль, горизонталь*.

Приятным сюрпризом служит тот факт, что маркеры расставляются в дракон-алгоритме не вручную, а автоматически. Эту операцию выполняет программа дракон-конструктор.

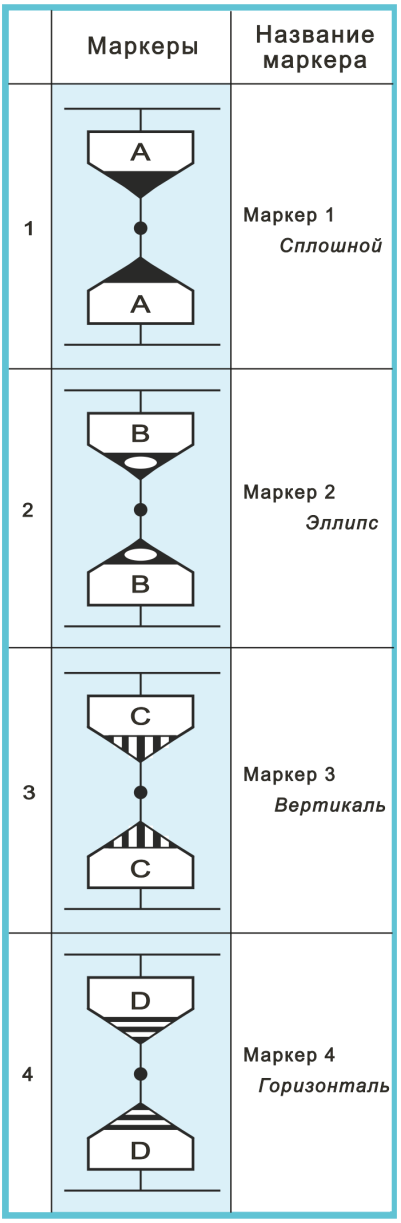


Рис. 22. Маркеры нужны для того, чтобы легко различать веточные циклы при зрительном восприятии

ДВА ЯЗЫКА

У медицинского ДРАКОНа есть старший брат, ракетно-космический ДРАКОН. Последний значительно богаче медицинского, у него вдвое больше икон и макроикон. Так что, если медикам понадобятся дополнительные

выразительные средства, можно заимствовать нужные иконы у старшего брата.

ВЫВОДЫ

1. Графические фигуры ДРАКОНа делятся на три части:
   - Иконы.
   - Макроиконы.
   - Маркеры.
2. Иконы и макроиконы – это мелкие блоки, из которых строятся дракон-алгоритмы.
3. Маркеры никак не влияют на правильность алгоритмов. Они облегчают чтение и зрительное восприятие наиболее сложных графических чертежей.
4. Данная глава представляет собой справочник. При анализе рисунков читатель может спросить, как называется та или иная фигура. Глава позволит быстро разобраться и сэкономить драгоценное время.

Глава 7

ПРОСТЫЕ МЕДИЦИНСКИЕ АЛГОРИТМЫ. ПРАВИЛА И ПРИМЕРЫ

ПРИМЕР МЕДИЦИНСКОГО АЛГОРИТМА

Проанализируем алгоритм на рис. 23. В верхней части в иконе Заголовок написано название алгоритма: «Реанимационные действия при наличии у новорожденного кистозной гигромы».

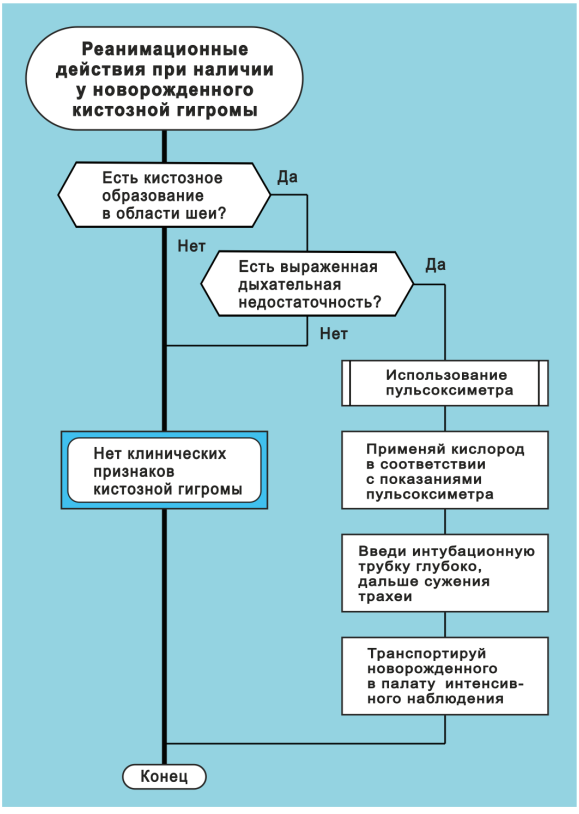


Рис. 23. Алгоритм «Реанимационные действия при наличии у новорожденного кистозной гигромы» [226]

Чтобы прочитать алгоритм, надо проследить все тропинки, ведущие из Заголовка в Конец, и понять их смысл.

В данном алгоритме встречаются иконы: Действие, Вопрос, Вставка, Комментарий (см. подсказки на рис. 20).

Найдите три иконы Действие:

- «Применяй кислород в соответствии с показаниями пульсоксиметра».
- «Введи интубационную трубку глубоко, дальше сужения трахеи».
- «Транспортируй новорожденного в палату интенсивного наблюдения».

Прочитайте надписи в иконах Вопрос:

- «Есть кистозное образование в области шеи?»
- «Есть выраженная дыхательная недостаточность?»

Икона Комментарий украшена синей каемкой. Она содержит важное пояснение:

- «Нет клинических признаков кистозной гигромы.

ИКОНА «ВСТАВКА»

Иногда бывает так, что на чертеже места уже нет, а работа все еще не закончена. Автор хочет любой ценой вставить в чертеж дополнительный кусок алгоритма. А места совсем нет. Как же быть?

Все очень просто. Не поместившийся кусок надо нарисовать в другом месте. А здесь, словно памятный знак, поместить икону Вставка. Она будет напоминать, что часть алгоритма «переехала» на новое место.

На рис. 23 икона Вставка снабжена надписью: «Использование пульсоксиметра». Это значит, что алгоритм с таким названием нарисован в другом месте, на рис. 24.

Слово «вставка» означает, что икону Вставка можно мысленно удалить, а вместо нее вставить нужный алгоритм.

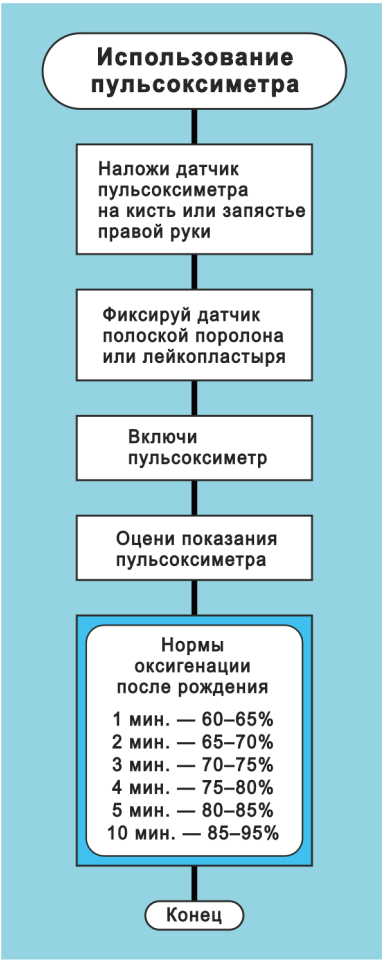


Рис. 24. Алгоритм «Использование пульсоксиметра» [227]

Применительно к нашему случаю можно мысленно удалить икону «Использование пульсоксиметра» на рис. 23, а вместо нее вставить алгоритм, изображенный на рис. 24.

Назначение Вставки в том, чтобы вынести из большого алгоритма фрагмент, решающий отдельную задачу и требующий пристального внимания. Можно сказать по-другому. Вставка представляет собой икону-заместитель. Она *замещает* отсутствующий алгоритм и информирует, что на данном рисунке вы его не найдете; он «уехал» на другой участок.

ЧТО ТАКОЕ МАРШРУТ

Понятие «маршрут» нужно для того, чтобы создать в алгоритме образцовый порядок. Дракон-алгоритм (рис. 23) имеет одно начало и один конец. *Маршрут* – путь, ведущий из начала в конечный пункт алгоритма. Все дракон-маршруты можно проследить пальцем от начала до конца, не отрывая палец от бумаги. Чертеж позволяет врачу одновременно видеть интересующие его маршруты.

Алгоритм на рис. 23 имеет три маршрута. Все они начинаются в иконе Заголовок и кончаются в иконе Конец.

Язык ДРАКОН предъявляет врачу в качестве подсказки полный обзор алгоритмической ситуации, т. е. полный набор маршрутов.

Правило Дракона

- Любой дракон-алгоритм имеет только одно начало (один вход). И только один конец (один выход).
- Запрещено иметь в алгоритме несколько концов (несколько выходов).

Что такое маршрут

Это путь, идущий из начала алгоритма (из иконы Заголовок) до конца алгоритма (до иконы Конец).

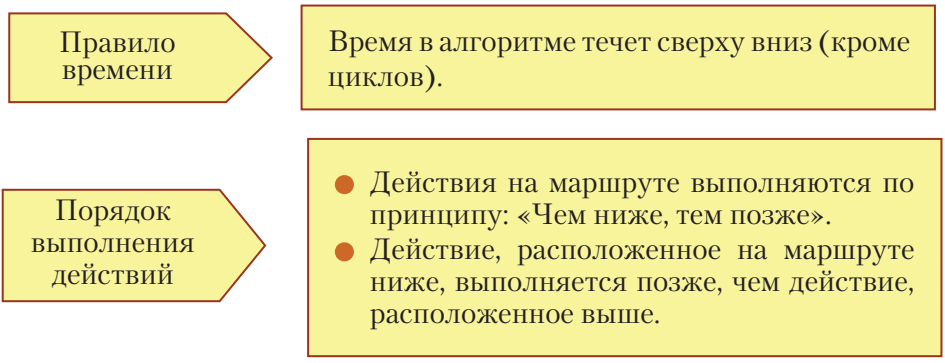
ЧТО ЛУЧШЕ: ПОРЯДОК ИЛИ ПУТАНИЦА?

Чем хороша географическая карта? Тем, что она упорядочена. Вверху север, внизу юг. Смотрим вверх – видим север, смотрим вниз – видим юг. Двигаясь по меридиану сверху вниз, мы перемещаемся с севера на юг.

Важнейшая цель дракон-алгоритма – *упорядочить* алгоритмическую картину болезни и устранить путаницу. Для этого он выстроен по образцу географической карты.

Дракон-алгоритм на рис. 23 нарисован не как попало, а по строгим правилам. Вверху врач видит начало алгоритма и начало времени. Внизу – конец алгоритма и конец времени. Двигаясь по алгоритму сверху вниз, мы перемещаемся во времени от начального момента до конечного.

Чтобы обеспечить в алгоритме дисциплину и порядок, введена специально разработанная эргономичная система понятий и правил [71].



ВРЕМЯ ТЕЧЕТ СВЕРХУ ВНИЗ

Вертикаль играет важную роль в дракон-алгоритме. Время направлено вертикально вниз. На рис. 23 мы видим три вертикальных линии. На всех линиях время течет вниз.

«Чем ниже, тем позже». Действие «Транспортируй новорожденного...» выполняется после действия «Введи интубационную трубку...».

Благодаря использованию «вертикального времени» отпадает необходимость использовать стрелки.

Чтобы облегчить понимание алгоритма, вводится понятие бегунка. *Бегунок* – воображаемая точка, которая последовательно пробегает все иконы одного из маршрутов, перемещаясь из начала в конец.

ГЛАВНЫЙ МАРШРУТ И ШАМПУР

Главный маршрут медицинского алгоритма – наиболее желательный и благоприятный для больного путь от иконы Заголовок до иконы Конец.

Даже если алгоритм описывает печальные варианты, включая летальный исход, главный маршрут всегда описывает наиболее *благоприятный для пациента* вариант (из числа возможных).

На рис. 23 и 24 главный маршрут показан жирной линией.

Шампур – вертикальная прямая линия, соединяющая иконы Заголовок и Конец. Между ними на той же линии помещается одна или несколько других икон.

ПРАВИЛО ГЛАВНОГО МАРШРУТА

Рассмотрим задачу. В запутанном лабиринте разветвленного медицинского алгоритма, нужно выделить один-единственный маршрут – царскую дорогу, путеводную нить (рис. 25). С ней можно зрительно сравнивать остальные маршруты, чтобы понять суть дела и не заблудиться в паутине развилок и тропинок.

Путеводная нить (главный маршрут) должна быть легко различима, она должна бросаться в глаза. Бросив беглый взгляд на дракон-алгоритм, врач сразу же видит царский маршрут и упорядоченные относительно него остальные маршруты.

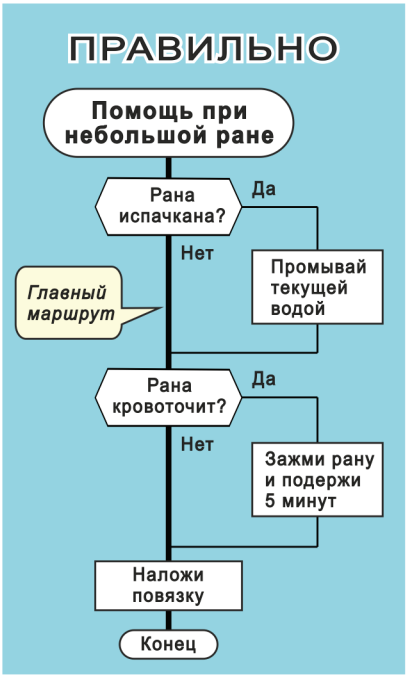


Рис. 25. Главный маршрут показан жирной линией. Правило главного маршрута выполняется. Поэтому царский путь прямой как стрела [228]

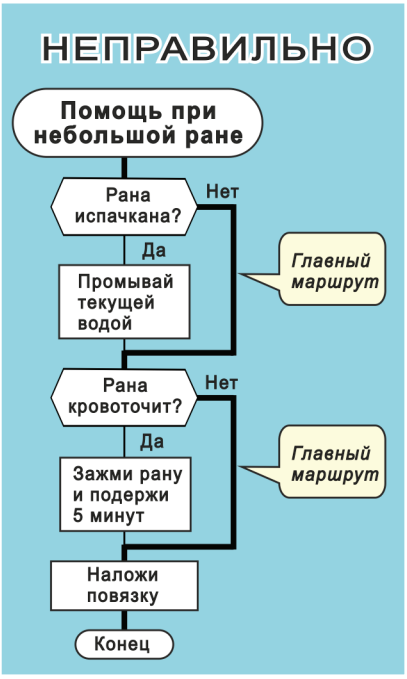


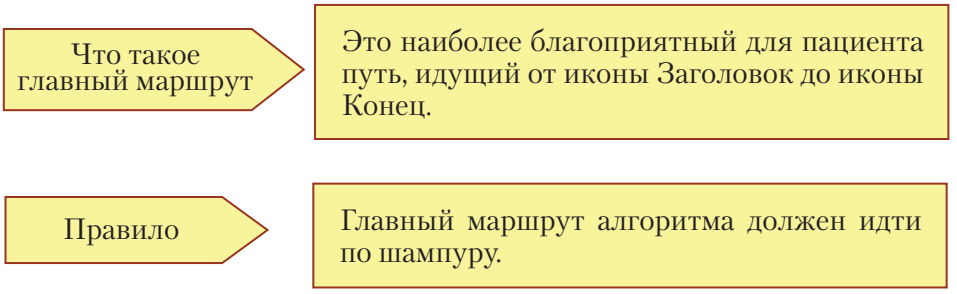
Рис. 26. Правило нарушено и главный маршрут «испортился», стал изломанным

Приятной новостью для медиков служит правило: *«Главный маршрут алгоритма должен идти по шампуру»*.

Это значит, что царская дорога не может оказаться где-то на задворках медицинского алгоритма, где ее днем с огнем не сыскать. Нет, она всегда должна быть на самом почетном месте – на крайней левой вертикали.

Почему? Потому что крайняя левая вертикаль описывает самый хороший, самый предпочтительный для больного вариант развития заболевания или хирургической операции

Подобный порядок очень удобен. Он делает дракон-алгоритм четким, предсказуемым и интуитивно понятным.



ИСПОРЧЕННЫЙ ГЛАВНЫЙ МАРШРУТ

Рассмотрим типичную ошибку. Предположим, правило главного маршрута не соблюдается (рис. 26).

Давайте проверим условие: «Рана испачкана?». Если рана чистая, это хорошо, если грязная – плохо. Главный маршрут всегда идет там, где хорошо. Следовательно на рис. 26, в развилке «Рана испачкана?», главный маршрут проходит через Нет, т. е. сворачивает с вертикали.

Отвечаем на второй вопрос: «Рана кровоточит?». Если крови нет, это хорошо, а если есть кровотечение – плохо. Главный маршрут любит, где хорошо. Поэтому, в развилке «Рана кровоточит?», он идет через Нет.

Получается, что на рис. 26 царский путь два раза отклоняется от шампура и начинает делать зигзаги. Это недопустимо.

На рис. 25 ошибка исправлена – главный маршрут идет по шампуру.

ВРАЧ ОБЯЗАН ЗНАТЬ ВСЕ МАРШРУТЫ АЛГОРИТМА

Рассмотрим простой вопрос. Сколько маршрутов на рисунке 25? Ответ: четыре. Все они показаны в таблице ниже.

Таблица правильно отражает ситуацию, но у нее есть недостаток – она не наглядна. Гораздо лучше и отчетливее видны маршруты на рис. 27. Левая картинка показывает, что главный маршрут прямой как стрела. Остальные три маршрута боковые. Они расположены правее шампура на одном или двух участках.

Обратите внимание: на рис. 27 все иконы Вопрос «одноногие»; у них показан только один выход. Это сделано не случайно.

	Два ответа в верхней и нижней развилках	**Как идет маршрут?**
Маршрут 1 *(главный)*	**Нет Нет**	Маршрут идет по шампуру. Это главный маршрут, когда все благополучно (рана чистая и нет кровотечения).
Маршрут 2 *(боковой)*	**Нет Да**	В верхней развилке маршрут сворачивает направо через Да (рана грязная), а в нижней идет вниз через Нет (нет кровотечения).
Маршрут 3 *(боковой)*	**Да Нет**	В верхней развилке маршрут идет вниз через Нет (рана чистая), в нижней поворачивает направо через Да (рана кровоточит).
Маршрут 4 *(боковой)*	**Да Да**	В обеих развилках маршрут уходит вправо через Да. Это самый плохой вариант, когда рана грязная и кровоточит. Данный маршрут сдвинут вправо на обоих участках.

РАЗВЕРТКА АЛГОРИТМА

Зачем нужен рисунок 27? Он поясняет и разворачивает рис. 25. Чтобы выделить один маршрут из четырех на рис. 25, нужно в каждом разветвлении выбрать только один ответ, а второй отбросить и удалить.

Это значит, что мы *развернули* алгоритм на рис. 25, показали все дорожки по отдельности и расположили их по соседству (рис. 27). Полученный результат носит название *развертка алгоритма*. Она позволяет воочию увидеть все без исключения маршруты алгоритма.

Здесь есть одно но. Подобные развертки – слишком дорогое удовольствие. Чтобы построить развертку сложного алгоритма, нужно затратить много труда и времени. Это допустимо только как исключение и лишь для особо сложных случаев.

Во всех остальных случаях достаточно иметь обычный дракон-алгоритм (как на рис. 25) и уметь с ним работать, не прибегая к трудоемкой развертке.

На рис. 25 и 27 мы рассмотрели простейший разветвленный алгоритм. Тем не менее, тщательный анализ этого случая (при помощи развертки) оказывается далеко не простым.

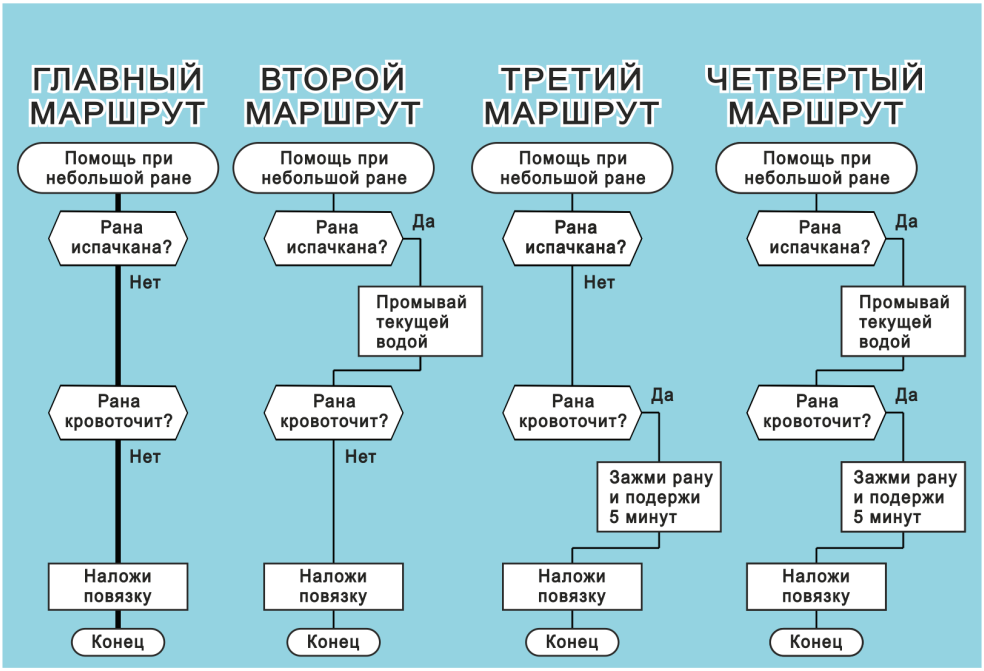


Рис. 27. Развертка алгоритма, представленного на рис. 25

Проблема в том, что простые алгоритмы в медицине встречаются редко. Реальные алгоритмы, как правило, очень сложны.

Неумение анализировать сложные алгоритмы и распознавать их маршруты является причиной многих врачебных ошибок.

АЛГОРИТМ УПОРЯДОЧЕН ПО ГОРИЗОНТАЛИ

Вспомним еще раз географическую карту. Слева запад, справа восток. Смотрим налево – видим запад, смотрим направо – видим восток. Движение взора по горизонтали упорядочено и имеет четкий смысл.

Дракон-алгоритм тоже упорядочен по горизонтали. Слева врач видит наиболее благоприятный для пациента маршрут, справа – наиболее тяжелый.

Двигаясь по алгоритму слева направо, мы перемещаемся от хорошей ситуации к плохой, от желательной – ко все более неприятной и угрожающей, вплоть до летального исхода.

Действует правило: «Слева хорошо, справа плохо», или «Слева лучше, справа хуже». Говоря более точно: «Слева более благополучный для больного исход, справа не благоприятный».

ПРАВИЛО БОКОВЫХ МАРШРУТОВ

Боковые маршруты нужно рисовать справа от шампура по принципу: *«Чем правее, тем хуже»*.

Дракон-алгоритм превращает хаос в порядок. Для наглядности развернем эту мысль в шутливой форме на рис. 28. Все маршруты упорядочены согласно правилу: «Чем правее расположен маршрут, тем более неприятную ситуацию он описывает».

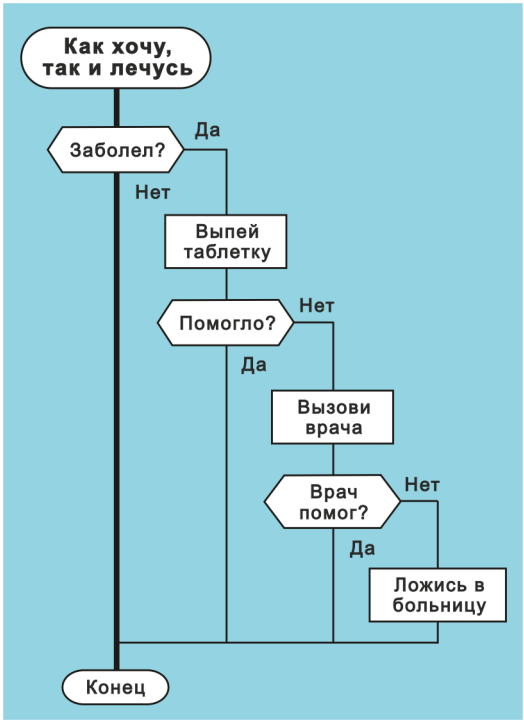


Рис. 28. Маршруты упорядочены слева направо

Левая вертикаль означает, что дела идут хорошо, ибо человек здоров. Вторая вертикаль описывает легкое недомогание, которое можно снять таблеткой. Третья вертикаль говорит: самочувствие ухудшилось, нужен врач. Наконец, крайняя правая вертикаль отражает самую неприятную ситуацию – пришлось лечь в больницу.

Что такое боковой маршрут	Это любой маршрут разветвленного алгоритма за исключением главного.
Правило боковых маршрутов	Боковые маршруты алгоритма нужно рисовать справа от шампура по принципу: «Чем правее, тем хуже».

КАРТОГРАФИЧЕСКИЙ ПРИНЦИП ЯЗЫКА ДРАКОН

Мы неоднократно подчеркивали, что дракон-алгоритм похож на географическую карту. Введем термин «Картографический принцип языка ДРАКОН». Принцип означает, что движение взора по горизонтали имеет стро-

го определенный смысл. Слева находятся хорошие (более благоприятные для пациента) маршруты, справа – плохие (менее благоприятные).

Точно так же перемещение взгляда по вертикали имеет ясный смысл: вверху начало времени, внизу – конец.

Картографический принцип – обобщающее и емкое понятие, которое включает в себя все понятия и правила, направленные на устранение визуальной путаницы и вводящие в дракон-алгоритм мудрый порядок и железную дисциплину. Сюда относятся: правило шампура, правило главного маршрута, правило боковых маршрутов, правило времени и т. д.

ЧТО ТАКОЕ ПЕРЕКЛЮЧАТЕЛЬ

Схема на рис. 29 позволяет сделать в алгоритме развилку на три направления. Для этого используются две иконы Вопрос. Однако задачу можно решить и по-другому – с помощью переключателя (рис. 30).

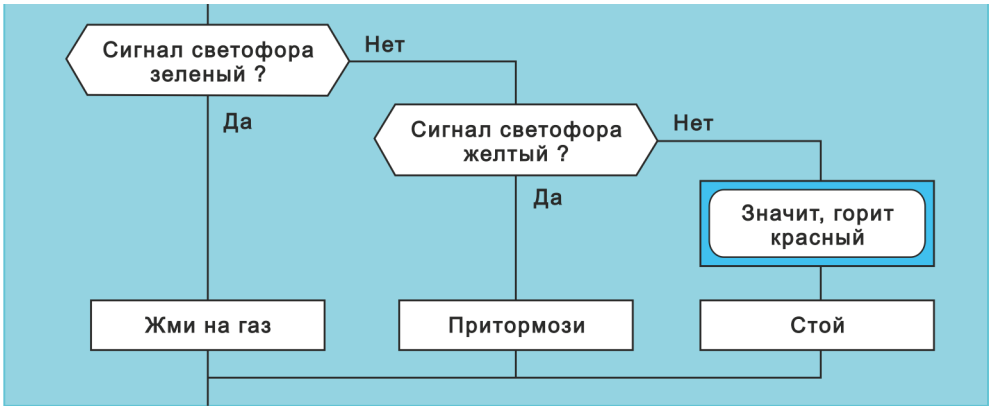


Рис. 29. Развилка на три направления, построенная с помощью иконы Вопрос

Переключатель – разветвление алгоритма на несколько тропинок, которые потом сливаются в одну. Переключатель сложная структура. Она строится из простых кирпичиков. Такими кирпичиками служат иконы Выбор и Вариант. Зачем они нужны?

Икона Выбор содержит вопрос или приказ, имеющий несколько ответов. Каждый ответ пишут в отдельной рамочке – иконе Вариант.

Взглянем на рис. 30. На первый взгляд там нет вопроса. Однако на самом деле вопрос есть, правда неявный. Чтобы убедиться, слово «светофор» прочитаем так: «Какой сигнал светофора сейчас горит?». Получим три ответа:

- зеленый,
- желтый,
- красный.

Попробуем описать, как работает переключатель. Описание обычно начинается со слова «если». Вот примеры.

- Если светофор зеленый – жми на газ.
- Если светофор желтый – притормози.
- Если светофор красный – стой (рис. 30).

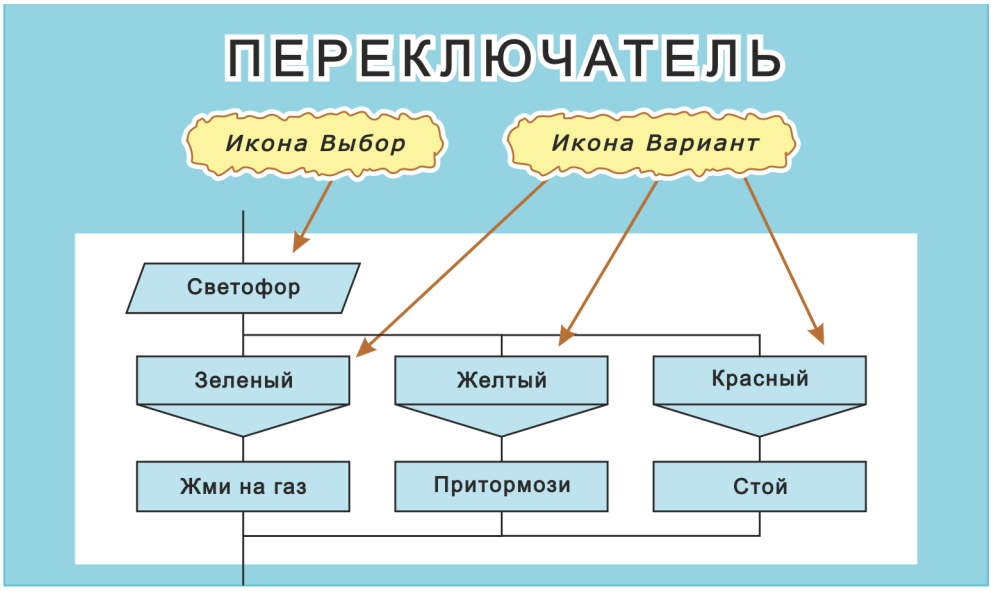
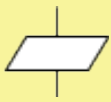


Рис. 30. Развилка на три направления, построенная с помощью переключателя

Что такое переключатель

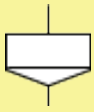
- Это часть алгоритма, имеющая один вход и один выход, внутри которой алгоритм разветвляется на несколько дорожек.
- Переключатель строится с помощью икон Выбор и Вариант.

Что такое икона Выбор



- Икона, которую рисуют в начале переключателя.
- В ней пишут вопрос или приказ, имеющий два и более ответов.

Что такое икона Вариант



Икона переключателя, в которой пишут один ответ на вопрос.

ПЕРЕКЛЮЧАТЕЛЬ ДЛЯ ВЫБОРА МЕДИЦИНСКОГО ИНСТРУМЕНТА

В медицинских алгоритмах часто встречаются переключатели. Сначала рассмотрим уже знакомый нам пример. Он извлечен из рис. 2 и перенесен на рис. 31.

Вверху в иконе Выбор дано подробное указание (приказ): «Выбери средство для освобождения дыхательных путей». Иконы Вариант поясняют, что таких средств два: интубационная трубка и катетер. Стало быть, надо выбрать одно из двух: либо трубку, либо катетер.

При желании в иконе Выбор текст можно изменить и написать кратко: «Трубка или катетер?»

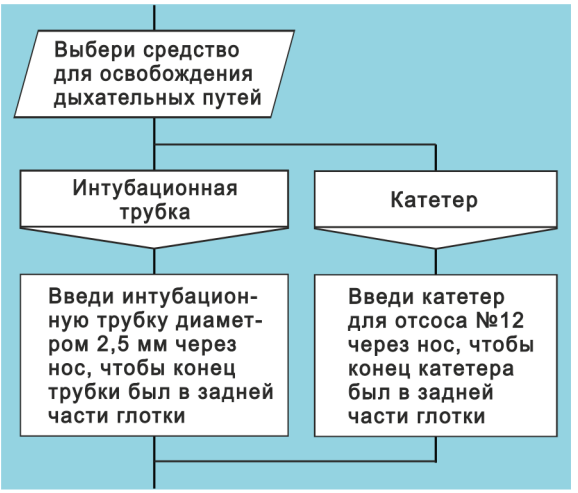


Рис. 31. Переключатель на два направления

Внизу на рис. 31 в иконах Действие указаны конкретные команды: «Введи интубационную трубку...» и «Введи катетер для отсоса...».

Сколько шагов изображено в алгоритме на рис. 31? Два шага. Первый шаг – нужно принять решение, что будем использовать: трубку или катетер? Второй шаг – выполнить физическое действие.

В качестве упражнения давайте мысленно удалим из рис. 31 два действия и вместо них на вертикальных линиях поместим два черных кружка. Зачем? Об этом речь пойдет в следующем параграфе.

И последнее. Какой из маршрутов на рис. 31 более благоприятен для пациента: левый или правый? Ответить затруднительно. Следовательно, правило «Чем правее, тем хуже» здесь не применимо. Это значит, что мы столкнулись с исключением, но так бывает редко.

В большинстве случаев принцип «Чем правее, тем хуже» эффективно работает и позволяет красиво расположить маршруты алгоритма. В этом можно убедиться при анализе примеров на рис. 32–34.

ПЕРЕКЛЮЧАТЕЛЬ И ТЯЖЕСТЬ ЗАБОЛЕВАНИЯ

Переключатель позволяет указать число маршрутов и помогает врачу выбрать один из них. Для удобства читателя мы рассматриваем простейший случай, когда маршрутов всего два.

Выше мы изучили переключатель для выбора медицинского инструмента. На рис. 32 у переключателя иная функция – он учитывает тяжесть заболевания и отделяет легкий химический ожог от средних и тяжелых. В соответствии с этим маршруты четко упорядочены слева направо, причем более тяжелый ожог расположен правее. Это значит, что выполняется правило боковых маршрутов «Чем правее, тем хуже».

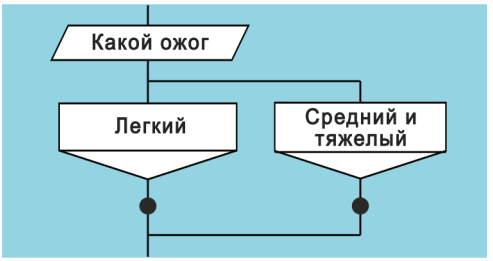


Рис. 32. Переключатель на два направления с валентными точками (черные кружки)

Обратите внимание: на рис. 32 показан только первый шаг (решение), а второй шаг (действие) скрыт и изображен условно, в виде черных кружков. Кружки обозначают валентные точки алгоритма. В эти точки чуть позднее мы введем иконы Действие или другие полезные вещи.

ПЕРЕКЛЮЧАТЕЛЬ И ПОРАЖЕННЫЕ ОРГАНЫ

Незнакомый алгоритм гораздо легче воспринимается, если читатель заранее знает, что слева находятся «хорошие» маршруты, а справа «плохие», неблагоприятные для пациента.

Вот пример. Ожог конъюнктивы и роговицы глаза опаснее, чем ожог века. Поэтому эти два случая необходимо упорядочить слева направо по степени тяжести.

Правило боковых маршрутов «Чем правее, тем хуже» является хорошим критерием для этого случая. Оно позволяет внести ясность в алгоритм, избежать путаницы и облегчить изучение материала.

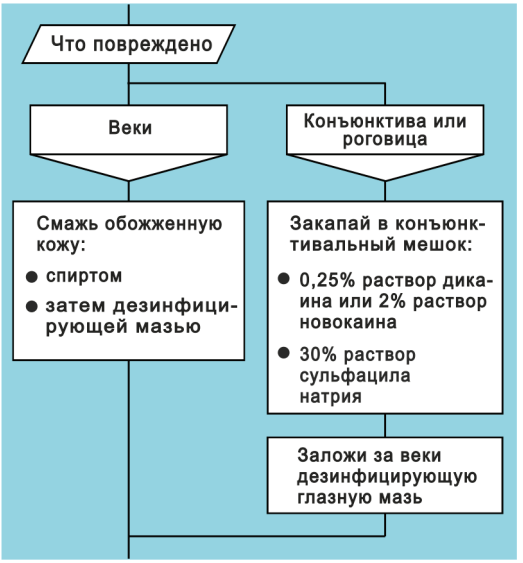


Рис. 33. Переключатель на два направления. Выполняется правило «Чем правее, тем хуже»

На рис. 33 в иконе Выбор записан вопрос: «Что повреждено?». В иконах Вариант дан ответ: «Веки» и «Конъюнктива...». Ниже в трех иконах Действие описаны соответствующие лечебные воздействия.

ДВА ПЕРЕКЛЮЧАТЕЛЯ В ОДНОМ АЛГОРИТМЕ

Перейдем к более сложному случаю и рассмотрим алгоритм «Первая помощь при химическом ожоге глаз жидкостью». Кусочки этого алгоритма мы уже видели на рис. 32 и 33.

Подправим наш эскиз, объединим кусочки и добавим последний штрих. Результат показан на рис. 34. Чтобы понять структуру нового алгоритма, подскажем: он построен с помощью двух переключателей.

Построение выполняется за три этапа.

Этап 1. Выбираем переключатель на рис. 32.

Этап 2. В левый черный кружок (ниже варианта «Легкий») вставляем переключатель из рис. 33.

Этап 3. В правый черный кружок вставляем три иконы Действие:

- «Введи подкожно...».
- «Введи внутиримышечно...».
- «Наложи на поврежденный глаз...».

Выполнив эти операции, получим окончательный алгоритм на рис. 34. Нетрудно заметить, что он состоит из трех маршрутов. Главный маршрут, как и полагается, идет по шампуру. Он обозначен жирной линией.

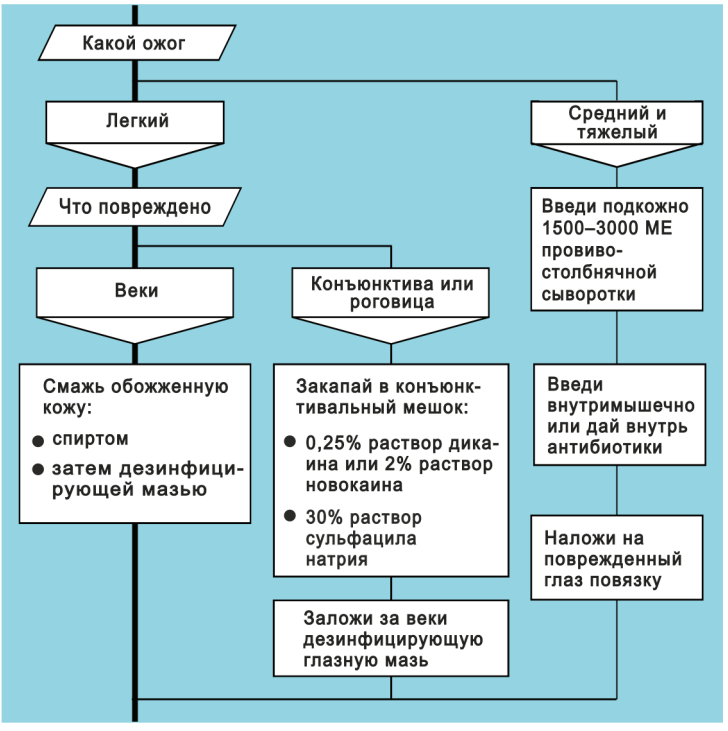


Рис. 34. Алгоритм с двумя переключателями

МАКРОИКОНА ПЕРЕКЛЮЧАТЕЛЬ

Теперь пора вспомнить про макроикону «Переключатель». В медицинском языке ДРАКОН имеется девять макроикон, которые показаны в справочнике на рис. 21. Переключатель занимает там почетное второе место.

Итак, мы убедились, что на рис. 34 показан фрагмент первой помощи при химическом ожоге глаза с использованием макроиконы «Переключатель».

Верхний переключатель на рис. 34 содержит вопрос: «Какой ожог»? Две иконы Вариант предлагают выбрать один из двух ответов:

- «Легкий».
- «Средний и тяжелый».

Нижний переключатель спрашивает: «Что повреждено»? В иконах Вариант читаем ответы:

- «Веки».
- «Конъюнктива или роговица».

ЧТО МЫ УЗНАЛИ В ЭТОЙ ГЛАВЕ

ДРАКОН – графический язык. Он имеет графические буквы. Необходимо запомнить их. Нужно знать, как они выглядят и как называются. Это нетрудно.

В этой и предыдущих главах мы познакомились с «золотой десяткой» графических фигур ДРАКОНа.

Проверьте себя. Закройте названия на рис. 35 рукой или газетой и, глядя на фигуры, постарайтесь вспомнить, как они называются. После этого просмотрите предыдущие рисунки (рис. 1 – 33) и найдите на них все 10 фигур.

Учтите, что в состав золотой десятки входят восемь икон (1-8) и две макроиконы (9 и 10).

Самые ходовые среди них – иконы Действие и Вопрос.

Рис. 35. Часто используемые фигуры в медицинских алгоритмах

ВЫВОДЫ

1. Дракон-алгоритм имеет одно начало и один конец.
2. Запрещено иметь в алгоритме несколько концов.
3. Маршрут – путь, идущий из иконы Заголовок до иконы Конец.
4. Главный маршрут – это наиболее благоприятный для пациента путь, соединяющий иконы Заголовок и Конец.
5. Шампур – вертикальная линия между Заголовком и Концом.
6. Шампур изображают жирной линией.
7. Главный маршрут алгоритма должен идти по шампуру.
8. Боковой маршрут – любой путь разветвленного алгоритма за исключением главного.
9. Боковые маршруты нужно рисовать справа от шампура по принципу: «Чем правее, тем хуже».
10. Пересечения линий запрещены.
11. Картографический принцип языка ДРАКОН делает движения взора (по горизонтали и вертикали) осмысленными. Этот принцип вносит в дракон-алгоритм строгую дисциплину и визуальный порядок.
12. Икона Вставка содержит название алгоритма и сообщает, что алгоритм с таким названием нарисован в другом месте.

Глава 8

ЛОГИКА В МЕДИЦИНЕ И НЕВИДИМАЯ МАТЕМАТИКА

КАК ПРЕВРАТИТЬ МЕДИЦИНСКИЙ ТЕКСТ В АЛГОРИТМ? НАДО УБРАТЬ ВСЕ ЛИШНЕЕ

В медицинской литературе алгоритмы обычно существуют в виде «вкраплений», которые перемешаны с другой информацией. Чтобы вычленить алгоритмические знания, их необходимо очистить и отделить от других тем и материалов. Грубо говоря, надо отделить алгоритмическое зерно от шелухи и мякины.

В данном случае другие темы (шелуха и мякина) – это различные декларативные медицинские знания, пояснения, обоснования, аргументация, доказательства, ссылки на научные эксперименты и т. д. Это, конечно, совсем не мякина, а исключительно ценные и важные сведения, но (!) совершенно бесполезные и даже вредные при рисовании алгоритмов. Поэтому при разработке медицинских алгоритмов все лишнее необходимо выделить, отсортировать и удалить.

Поясним сказанное на простом примере, относящемся к аутоиммунной гемолитической анемии.

> «При резистентности к кортикостероидам и внутривенному введению иммуноглуболина в последние годы ряд авторов рекомендуют применять ритуксимаб (мабтера) – химерные человеческие моноклональные антитела против *CD*20» [136].

Удалив из цитаты «шелуху и мякину», получим:

> «При резистентности к кортикостероидам и внутривенному введению иммуноглуболина... рекомендуют применять ритуксимаб...» [136].

Это и есть искомое алгоритмическое вкрапление, которое нас интересует.

КАК ПРЕВРАТИТЬ АЛГОРИТМИЧЕСКИЙ ТЕКСТ В ДРАКОН-АЛГОРИТМ

Мы сделали первый ход и получили словесное описание алгоритма. Теперь надо преобразовать его в графику. Это можно сделать двумя способами, как показано на рис. 36 и 37.

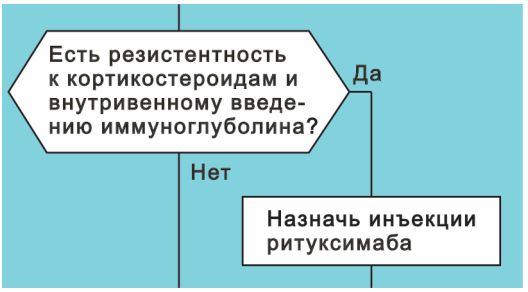


Рис. 36. Алгоритм лечения со сложным условием. Выход Нет *слева*

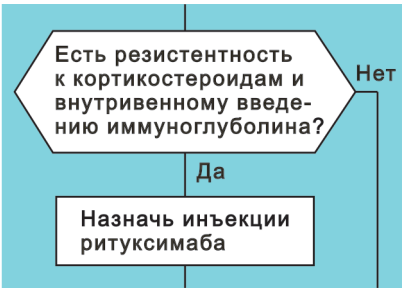


Рис. 37. Алгоритм лечения со сложным условием. Выход Нет *справа*

Подчеркнем: на обоих рисунках изображен один и тот же алгоритм. Он лишь нарисован по-разному. На первом рисунке выход Нет слева, на втором справа. Однако эта разница никак не влияет работу алгоритма.

Два алгоритма называются *равносильными*, если они имеют в точности одинаковые маршруты. Отсюда следует, что алгоритмы на рис. 36 и 37 равносильны.

В ДРАКОНе ЗАПРЕЩЕНЫ СЛОЖНЫЕ УСЛОВИЯ. ЧТО БУДЕМ ДЕЛАТЬ?

На рис. 36 и 37 в иконе Вопрос записано сложное логическое условие. Мы уже знаем, что так делать нельзя. Необходимо исправить ошибку, убрать сложное условие и заменить его на два простых. Результат показан на рис. 38 и 39.

На первый взгляд, между рисунками нет сходства. Однако это не так. Легко убедиться, что на рис. 38 и 39 присутствуют три совершенно одинаковых маршрута. Можно математически доказать, что алгоритмы на этих рисунках в точности совпадают, они равносильны[1].

Итак, мы выяснили, что два рисунка алгоритмически равны, но отличаются графически. Отличие связано со зрительным восприятием и эргономикой – левый рисунок эргономичнее и удобнее. Причина в том, что слева (на рис. 38) соблюдается картографический принцип ДРАКОНа, а справа – нет.

[1] Доказательство см. в работе [231].

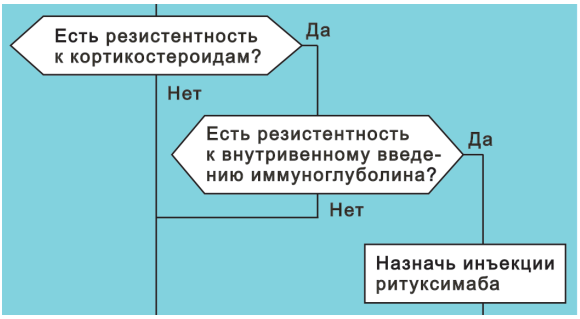


Рис. 38. Алгоритм лечения с двумя простыми условиями. Выход Нет *слева*. Картографический принцип соблюдается [136]

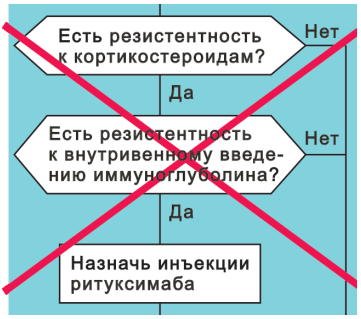


Рис. 39. Алгоритм лечения с двумя простыми условиями. Выход Нет *справа*. Картографический принцип нарушен

Поясним. Левая вертикаль на рис. 38 описывает более благоприятный для пациента маршрут (поскольку резистентность к кортикостероидам отсутствует), а правая – менее благоприятный (приходится делать инъекции ритуксимаба). Следовательно, при движении взора слева направо выполняется правило: «Чем правее, тем хуже».

Картографический принцип обеспечивает эргономическую упорядоченность и концептуальное единство во всех частях дракон-алгоритма.

КАК ВЫЯВИТЬ ЛОГИЧЕСКИЕ ПРИНЦИПЫ. ОБСУЖДЕНИЕ МЕТОДИКИ

Наша цель в этой главе – выявить Логику (логические принципы) РАЗВЕТВЛЕННЫХ медицинских алгоритмов, содержащих СЛОЖНЫЕ логические условия. Чтобы решить задачу, мы избрали следующий путь.

1. Найти в медицинской литературе подходящий пример в виде одного или двух предложений, которые содержат алгоритмическое вкрапление, которое должно иметь в своем составе:
   - разветвленный алгоритм,
   - сложное логическое условие.
2. Освободить выбранное предложение от посторонней информации, убрать лишние слова и вычленить алгоритмическую часть в чистом виде.
3. Преобразовать алгоритм из текстовой формы в пару равносильных графических алгоритмов. В обоих алгоритмах сложное логическое условие следует поместить в икону Вопрос, как на рис. 36, 37.
4. Преобразовать сложное логическое условие в несколько простых условий, причем каждое простое условие следует записать в отдельной иконе Вопрос (как на рис. 38, 39).

5. Выбрать из пары равносильных алгоритмов один, который удовлетворяет картографическому принципу языка ДРАКОН и признать его ***эргономичным и удобным для работы.*** А второй алгоритм забраковать и отбросить.

Пример, описанный в начале главы, полностью соответствует данной методике. Ниже представлены еще несколько иллюстраций.

ПРИМЕР 2. КАК ПРЕВРАТИТЬ МЕДИЦИНСКИЙ ТЕКСТ В ЭРГОНОМИЧНЫЙ АЛГОРИТМ

Рассмотрим еще одно алгоритмическое «вкрапление», относящееся к хронической обструктивной болезни легких (ХОБЛ).

> «Ингаляционные глюкокортикостероиды и ингаляционные глюкокортикостероиды в комбинации с бронхолитиками длительного действия назначаются при тяжелом течении ХОБЛ, при частых обострениях ХОБЛ – с целью снижения частоты обострений и улучшения качества жизни больных ХОБЛ» [137].

Слова «с целью снижения частоты обострений...» являются хорошим пояснением, но в алгоритме они неуместны. Алгоритм не научный трактат, здесь нельзя растекаться мыслью по древу. В алгоритмах полагается каленым железом выжигать шелуху и мякину.

Удалив все ненужное, получим:

> «Ингаляционные глюкокортикостероиды и ингаляционные глюкокортикостероиды в комбинации с бронхолитиками длительного действия назначаются при тяжелом течении ХОБЛ, при частых обострениях ХОБЛ» [137].

Преобразуем последний текст в графику. Как и раньше, сделаем это двумя способами и получим пару равносильных алгоритмов (рис. 40 и 41).

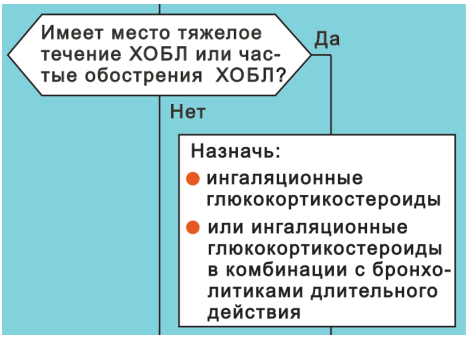


Рис. 40. Алгоритм ХОБЛ со сложным условием. Выход Нет *слева*

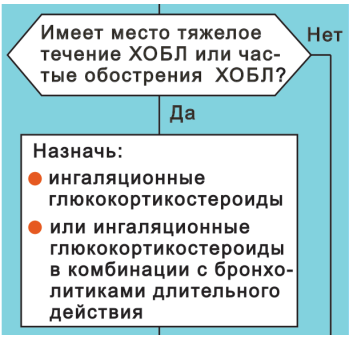


Рис. 41. Алгоритм ХОБЛ со сложным условием. Выход Нет *справа*

Согласно правилам ДРАКОНа заменим сложное условие на простое. Два искомых равносильных алгоритма показаны на рис. 42 и 43. Первый является хорошим и эргономичным (картографичным). А второй плохим, неудобным и ненужным.

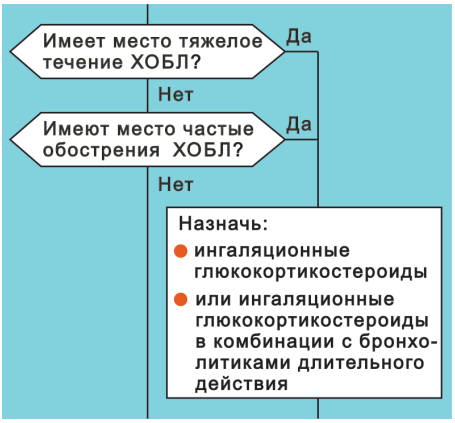


Рис. 42. Алгоритм ХОБЛ с двумя простыми условиями. Выходы Нет *слева*. Картографический принцип соблюдается [137]

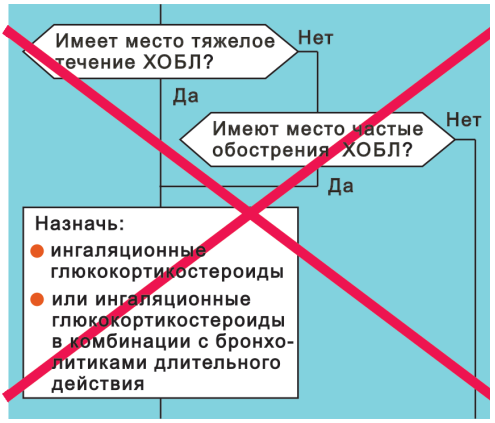


Рис. 43. Алгоритм ХОБЛ с двумя простыми условиями. Выход Нет *справа*. Картографический принцип нарушен

ПРИМЕР 3. КАК ПРЕВРАТИТЬ СЛОЖНЫЙ МЕДИЦИНСКИЙ ТЕКСТ В АЛГОРИТМ

Рассмотрим более сложную алгоритмическую логику, связанную с транзиторными ишемическими атаками. Как всегда, начнем с цитаты:

> «Выявление клинических симптомов транзиторных ишемических атак (онемение лица, половины тела, руки, ноги, кратковременные эпизоды выпадения зрения, эпизоды онемения языка, эпизоды головокружений), нарастающей церебральной недостаточности (снижения работоспособности, появление нарушений речи, снижение качества мыслительных процессов, другие когнитивные нарушения), появление шума в проекции сонных артерий при аускультации шеи, должно быть основанием направления больного для проведения ультразвукового допплерографического исследования состояния кровотока магистральных артерий головы. При выявлении стеноза пациент должен быть (!!!) направлен к сосудистому хирургу для решения вопроса о тактике оперативного или медикаментозного лечения» [138].

Удалив пояснения и лишние слова, получим концентрированную мысль:

> «Выявление клинических симптомов транзиторных ишемических атак..., нарастающей церебральной недостаточности..., появление шума в проекции сонных артерий при аускультации шеи, должно быть основанием для... ультразвукового допплерографического исследования состояния кровотока магистральных артерий головы. При выявлении стеноза пациент должен быть... направлен к сосудистому хирургу для решения вопроса о тактике оперативного или медикаментозного лечения» [138].

Преобразуем последний текст в графику. Действуя, как раньше, по накатанной колее, получим пару равносильных алгоритмов (рис. 44 и 45).

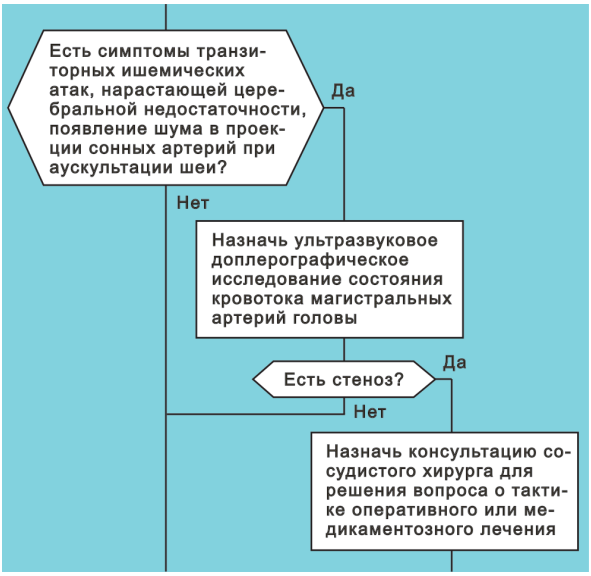


Рис. 44. Алгоритм со сложным условием. Выход Нет *слева*. Картографический принцип соблюдается

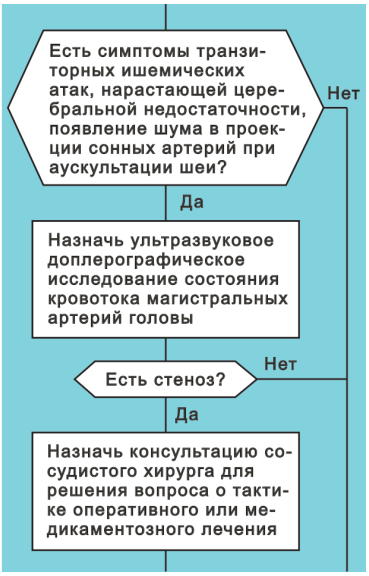


Рис. 45. Алгоритм со сложным условием. Выход Нет *справа*. Картографический принцип нарушен

Сделаем последнее улучшение – разобьем сложное условие на мелкие порции. В итоге увидим два равносильных алгоритма на рис. 46 и 47. Первый является хорошим и упорядоченным (картографичным), второй не удовлетворяет требованиям и отбрасывается.

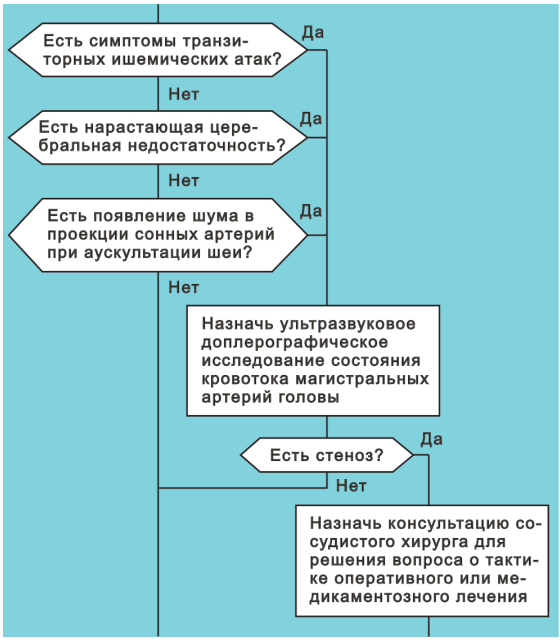


Рис. 46. Хороший алгоритм. Он удовлетворяет всем требованиям [138]

Рис. 47. Плохой алгоритм. Требования не выполнены, так как картографический принцип нарушен

КАК ПОМОЧЬ СТУДЕНТАМ ИЗУЧАТЬ МЕДИЦИНУ

В предыдущих параграфах мы рассмотрели проблему алгоритмических вкраплений, со всех сторон окруженных посторонней информацией. Парадокс в том, что окружающая медицинская информация сама по себе очень важна, но при описаниях медицинских алгоритмов она сильно вредит делу.

Алгоритмические вкрапления похожи на множество крошечных островков в безбрежном океане медицинской литературы. Между островками должны быть соединительные алгоритмические мостики, но они, как правило, лишь подразумеваются и явном виде не описаны. *Это фундаментальный недостаток мировой медицинской литературы.*

Подавляющее число вкраплений существуют в медицинских публикациях в виде изолированных точек. Они остаются одинокими алгоритмическими сиротами, не объединяются в логически законченные фрагменты и не превращаются в целостные алгоритмы.

Разумеется, студенты-медики и врачи в конце концов узнают многие алгоритмические секреты, используя разные источники и собственный

практический опыт. Но такое обучение нерационально и ресурсорасточительно. На подобное обучение люди вынуждены затрачивать неоправданно много времени. Иной раз почти всю жизнь. Можно ли сэкономить и сократить время, расходуемое на приобретение профессиональных медицинских знаний?

Отсутствие удобных, легко воспринимаемых эргономичных графических алгоритмов высокой точности в учебниках, стандартах, руководствах, клинических рекомендациях, протоколах до добра не доводит. Оно приводит к тому, что тяжкое бремя познания процедурных медицинских знаний взваливается на самих учащихся – на студентов-медиков и слушателей системы последипломного медицинского образования.

Сказанное означает, что проблема алгоритмизации медицинской литературы не только не решена, но даже не осознана.

Акцентируя внимание на недостатках, можно сказать, что современная медицинская литература не упорядочена. Она представляет собой эклектическую смесь (хаотическую мешанину) из декларативных и процедурных знаний.

Первоочередная задача состоит в том, чтобы:

- четко разграничить два типа знания,
- объединить алгоритмические вкрапления с помощью языка ДРАКОН и преобразовать их сначала в законченные алгоритмы, а затем в алгоритмические системы и комплексы.

В данной главе мы делаем лишь самый первый, предварительный шаг и решаем ограниченную задачу. С помощью двенадцати наглядных рисунков (рис. 36–47) мы проанализировали ситуацию и показали, что язык ДРАКОН позволяет преобразовать элементарные текстовые «вкрапления» в графическую форму, упорядочить графику с помощью когнитивно-эргономических методов. И за счет этого существенно облегчить восприятие, осмысление и понимание простейших алгоритмов.

На этом мы завершаем анализ проблемы текстовых алгоритмических вкраплений и переходим к исследованию и совершенствованию логики графических медицинских алгоритмов.

КРИТИКА И ИСПРАВЛЕНИЕ БЛОК-СХЕМЫ АЛГОРИТМА

Для наших целей удобно одновременно рассматривать пару равносильных алгоритмов. Продолжим эту традицию.

На рис. 48 и 49 изображены два равносильных алгоритма, нарисованные согласно стандарту на блок-схемы ГОСТ 19.701-90.

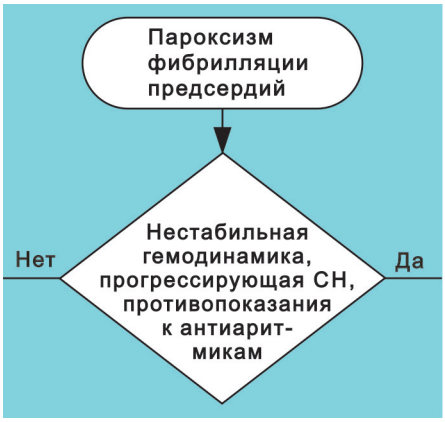


Рис. 48. Ромб и сложное условие. Выход Нет *слева*

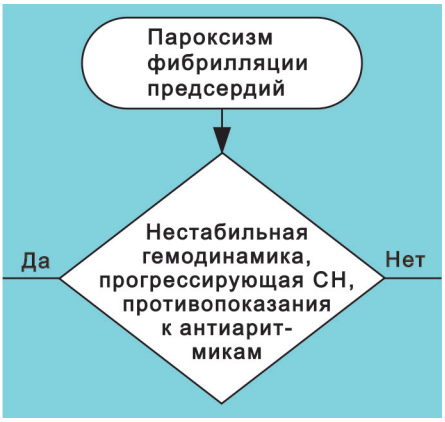


Рис. 49. Ромб и сложное условие. Выход Нет *справа*

С точки зрения когнитивной эргономики, можно отметить ряд недостатков, характерных для блок-схем.

1. Для обозначения развилки используется ромб, который занимает слишком много места и содержит мало текста. Ромб не позволяет поместить внутри необходимое количество удобочитаемого текста, состоящего из строк равной длины. Верхний и нижний углы ромба пропадают впустую, так как в них ничего нельзя записать.

   > В языке ДРАКОН ромбы запрещены, вместо них используется экономичная икона Вопрос. У нее верхний и нижний углы ромба «отпилены». Поэтому схема становится компактной и удобной как для записи текста, так и для чтения.

2. Блок-схема не упорядочена. Картографический принцип к ней не применим. Соединительные линии могут быть направлены в любую сторону. По этой причине в блок-схеме, чтобы указать направление движения бегунка по маршруту, необходимо рисовать множество стрелок, которые засоряют схему.

   > Дракон-схема строго упорядочена, она опирается на картографический принцип. Поэтому отпадает необходимость использовать стрелки (они нужны только в циклах).

ПРОДОЛЖЕНИЕ КРИТИКИ

Возвратимся к медицине. На рис. 49 изображен фрагмент алгоритма из учебника «Поликлиническая терапия», представленный в виде блок-схемы [139].

Внутри ромба записано трудное для понимания сложное условие, которое является скрытым источником врачебных ошибок. Преобразуем ромб в набор простых условий, причем сделаем это двумя способами – в виде двух равносильных алгоритмов, как показано на рис. 50 и 51.

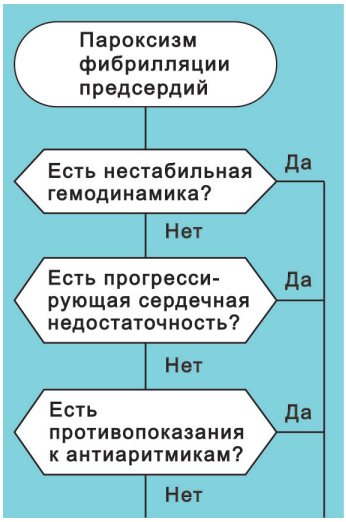


Рис. 50. Картографический принцип соблюдается [139]

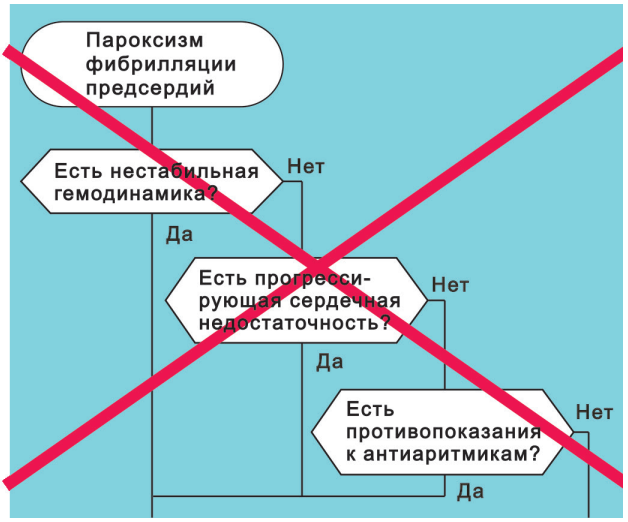


Рис. 51. Картографический принцип нарушен

Сравнивая их, можно заметить, что картографический принцип выполняется лишь слева, на рис. 50. Только этот алгоритм можно признать *эргономичным и удобным для работы*. Почему?

Ответить нетрудно. Все три условия («Есть нестабильная гемодинамика?» и остальные) носят негативный характер, они неблагоприятны для пациента. Чтобы отвести беду от больного, надо устранить негативные моменты и выбрать наилучший для него маршрут.

Для этого на все три негативных вопроса следует ответить Нет. Это и будет хороший маршрут, описывающий благоприятную для пациента ситуацию. На рис. 50 данный маршрут идет по шампуру, он является главным. Любое отклонение вправо от главного маршрута неблагоприятно и уводит нас на плохой маршрут. Следовательно, алгоритм на рис. 50 удовлетворяет правилу ДРАКОНа «Чем правее, тем хуже». Алгоритм на рис. 51 таким свойством не обладает.

НЕГАТИВНЫЕ И ПОЗИТИВНЫЕ ВОПРОСЫ

В этой главе мы много раз упражнялись, преобразуя сложные условия в простые. Перечислим наши упражнения.

- На рис. 38 и 39 показаны два простых условия (две иконы Вопрос).
- На рис. 42 и 43 также нарисованы две иконы Вопрос.
- На рис. 46 и 47 изображены уже не два, а три простых условия (три иконы Вопрос).
- На рис. 50 и 51 – тоже три иконы Вопрос.

А теперь самое любопытное. Оказывается, во всех случаях мы думали о плохом, а не о хорошем.

Мы нарочно использовали только *негативные* вопросы, т. е. вопросы, которые при ответе Да, указывают на неблагоприятную для пациента ситуацию. Например, «Есть прогрессирующая сердечная недостаточность?». Если есть, это плохо для пациента.

Пришло время перейти к анализу *позитивных* вопросов. Позитивный вопрос при ответе Да указывает на благоприятное для больного положение дел.

ЛОГИЧЕСКАЯ СХЕМА «ИЛИ»

Рассмотрим рис. 52 и 53. Пациент находится в угрожающем для жизни состоянии, решается вопрос о реанимации. В первую очередь врач должен ответить на вопросы:

- Есть ли реакция пациента на обращение и прикосновение?
- Есть ли нормальное (не агональное) дыхание?

Рис. 52. Схема «ИЛИ» с двумя условиями. Выход Да *слева*. Картографический принцип соблюдается

Рис. 53. Схема «ИЛИ» с двумя условиями. Выход Да *справа*. Картографический принцип нарушен

Если оба ответа «Нет», нужна реанимация, в противном случае – тщательный первичный осмотр.

С точки зрения медицины, реанимация – это главное. Однако у нас на уме другое – нас интересует логическая операция «ИЛИ». По этой причине забудем про реанимацию и переключим внимание на другое.

Попытаемся понять: в каком случае нужно проводить первичный осмотр пациента? Алгоритм на рис. 52 и 53 говорит, что это возможно в трех случаях.

1. Если есть реакция пациента на обращение и прикосновение. *(Маршрут выходит из верхней иконы Вопрос через Да)*.
2. Если есть нормальное (не агональное) дыхание. *(Маршрут выходит из верхней иконы Вопрос через Нет, затем пробегает через нижнюю икону Вопрос через Да)*.

   Это означает, что пациент не реагирует на обращение и прикосновение, однако дышит он нормально.

3. Если выполняются оба указанных условия.

   В этом случае состояние пациента благоприятное. Во-первых, он реагирует на обращение и прикосновение, во-вторых, у него нормальное дыхание.

Перечисленные соображения говорят о том, первичный осмотр пациента в алгоритме на рис. 52 и 53 выполняется по схеме ИЛИ.

Используя термин ИЛИ, ситуацию можно описать так. Первичный осмотр следует проводить в следующих случаях:

- ИЛИ пациент реагирует на обращение и прикосновение,
- ИЛИ пациент нормально дышит,
- ИЛИ имеют место оба этих признака.

Оба алгоритма на рис. 52 и 53 являются логически правильными. Оба реализуют схему ИЛИ. Разница лишь в том, что левый алгоритм соблюдает эргономические правила ДРАКОНа *(Главный маршрут должен идти по шампуру. Чем правее, тем хуже)*. И это хорошо.

А правый не соблюдает. И это плохо.

И последнее. На рис. 52 и 53 в иконах Вопрос мы впервые использовали позитивные вопросы. Этим они отличаются от рисунков 36–51, где повсеместно использовались негативные вопросы.

Как создать схему «ИЛИ»

- Соедините выходы «Да» двух икон Вопрос.
- Выход «Нет» верхней иконы Вопрос соедините со входом нижней иконы Вопрос.
- Объединенный выход «Да» двух икон Вопрос выполняет операцию «ИЛИ».

ЛОГИЧЕСКАЯ СХЕМА «ИЛИ» С ТРЕМЯ УСЛОВИЯМИ

Мы изучили схему ИЛИ с двумя условиями (с двумя иконами Вопрос). Рассмотрим такую же схему, но с *тремя* условиями. Возьмем за основу алгоритм на рис. 52, 53 и внесем в него нужные исправления.

Рассмотрим фразу «Есть ли реакция пациента на обращение и прикосновение?». При желании ее можно разделить на два вопроса. В некоторых случаях это бывает полезно. С учетом этого, взглянем на рис. 54, 55.

Рис. 54. Схема «ИЛИ» с тремя условиями. Выход Да *слева*. Картографический принцип соблюдается [160]

Рис. 55. Схема «ИЛИ» с тремя условиями. Выход Да *справа*. Картографический принцип нарушен

Мы видим, что врач должен ответить уже не на два, а на три отдельных вопроса:

- Есть ли реакция пациента на обращение?
- Есть ли реакция на прикосновение?
- Есть ли нормальное (не агональное) дыхание?

Если хотя бы на один вопрос получен ответ «Да», следует выполнить первичный осмотр (в противном случае нужна реанимация).

Обе схемы (рис. 54 и 55) логически правильны и равны друг другу. Какую из них следует предпочесть? Ту, которая удовлетворяет эргономичным правилам ДРАКОНа. В данном случае это левая схема ИЛИ. Правая схема нарушает требование «Чем правее, тем хуже» и отбрасывается.

Логическая операция ИЛИ означает: если хотя бы на один вопрос получен ответ «Да», выполни указанное в алгоритме действие (Первичный осмотр). Оба алгоритма (на рис. 54 и 55) равносильны и отличаются только способом начертания.

СХЕМА «ИЛИ» ДЛЯ ПОЗИТИВНЫХ И НЕГАТИВНЫХ ВОПРОСОВ

Мы познакомились с двумя новыми понятиями, или терминами:

- позитивный вопрос,
- негативный вопрос.

Им соответствуют две разные схемы:

- стандартная схема ИЛИ,
- нестандартная схема ИЛИ.

Все четыре примера на рис. 56–59 удовлетворяют правилам ДРАКОНа и являются правильными. Они рекомендуются в качестве образцов для подражания. Схема ИЛИ соответствует выходу Да.

Рис. 56. Стандартная схема «ИЛИ» с двумя позитивными вопросами [160]

Рис. 57. Нестандартная схема «ИЛИ» с двумя негативными вопросами [137]

На рис. 56 и 58 (где позитивные вопросы) шампур идет через Да. На рис. 57 и 59 (где негативные вопросы) шампур идет через Нет.

Стандартная схема ИЛИ похожа на «Лестницу с уступами», нестандартная – на «Мачту с парусами».

Рис. 58. Стандартная схема «ИЛИ» с тремя позитивными вопросами [160]

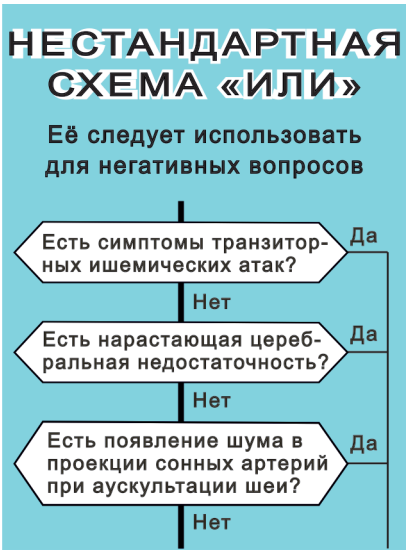


Рис. 59. Нестандартная схема «ИЛИ» с тремя негативными вопросами [138]

На рис. 56 лестница имеет два уступа, на рис. 58 – три. Точно так же на рис. 57 мачта имеет два паруса, на рис. 59 – три.

А если добавить еще один «этаж»? Лестница будет расти в ширину, а мачта – в высоту.

Правило 1 для схемы ИЛИ

Стандартная схема ИЛИ (лестница) предназначена для изображения двух или более *позитивных* вопросов.
При этом шампур помечают словом Да.

Правило 2 для схемы ИЛИ

Нестандартная схема ИЛИ (мачта) предназначена для изображения двух или более *негативных* вопросов.
При этом шампур помечают словом Нет.

ЛОГИЧЕСКАЯ СХЕМА «И» С ДВУМЯ УСЛОВИЯМИ

Схема «И» отличается тем, что необходимо совпадение двух или нескольких условий.

Предположим, надо повесить на стену картину. Для этого придется забить в стену гвоздь. Что для этого нужно? Нужен молоток и гвоздь. Если есть только гвоздь, а молотка нет, ничего не выйдет. И наоборот, если есть молоток без гвоздя, толку не будет. Непременно нужно и то и другое. Говорят, что нужно совпадение двух условий: и гвоздя, и молотка (рис. 60 и 61).

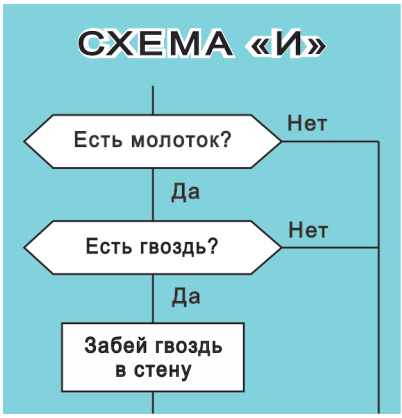


Рис. 60. Схема «И» с двумя условиями. Выход Да *слева*. Картографический принцип соблюдается

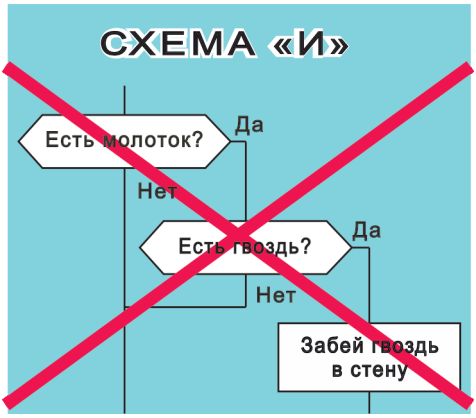


Рис. 61. Схема «И» с двумя условиями. Выход Да *справа*. Картографический принцип нарушен

Логическая операция И означает: если на оба вопроса получен ответ «Да», выполни действие, на которое указывает выход Да нижней иконы Вопрос (Забей гвоздь в стену). Оба алгоритма (на рис. 60 и 61) равносильны и отличаются только способом начертания.

ЛОГИЧЕСКАЯ СХЕМА «И». МЕДИЦИНСКИЙ ПРИМЕР

Рассмотрим равносильные алгоритмы на рис. 62 и 63. Речь идет об интубации трахеи. Предлагаются два вопроса для самоконтроля врача:

- Можешь ли провести интубацию?
- Больше ли пользы в интубации, чем риска?

Если оба ответа «Да», врач приступает к интубации.

Алгоритм на рис. 62 и 63 говорит, что проведение интубации трахеи возможно при совпадении двух условий.

1. Если врач обладает достаточным опытом и способен провести интубацию. (Маршрут выходит из верхней иконы Вопрос через Да).
2. Если врач уверен, что польза от интубации превышает возможный риск для пациента. (Маршрут выходит из нижней иконы Вопрос через Да).

Эти соображения говорят о том, интубация трахеи в алгоритме на рис. 62 и 63 выполняется по схеме «И».

Используя термин «И», ситуацию можно описать так. Интубацию трахеи следует проводить, если одновременно выполняются два условия:

- врач способен провести интубацию;
- и польза от интубации больше, чем риск.

Оба алгоритма на рис. 62 и 63 являются логически правильными. Оба реализуют схему «И». Разница лишь в том, что левый алгоритм соблюдает эргономические правила ДРАКОНа (Главный маршрут идет по шампуру. Чем правее, тем хуже).

А правый не соблюдает. Поэтому правый алгоритм считается негодным и отбрасывается.

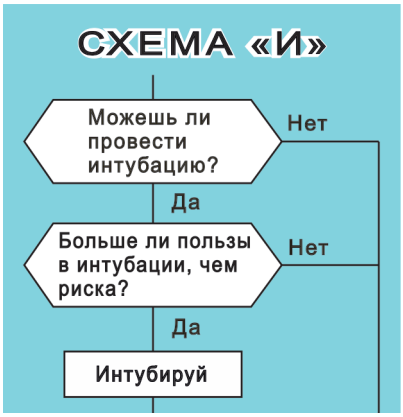


Рис. 62. Схема «И» с двумя условиями. Выход Да *слева*. Картографический принцип соблюдается [232]

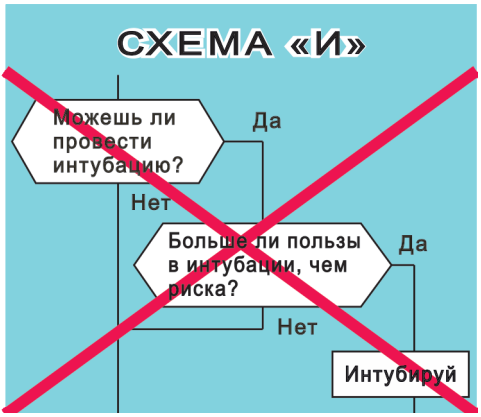


Рис. 63. Схема «И» с двумя условиями. Выход Да *справа*. Картографический принцип нарушен

Как создать схему «И»

- Соедините выход «Да» верхней иконы Вопрос со входом нижней иконы Вопрос.
- Соедините выходы «Нет» обеих икон Вопрос.
- Выход «Да» нижней иконы Вопрос выполняет операцию «И».

ЛОГИЧЕСКАЯ СХЕМА «И» С ТРЕМЯ УСЛОВИЯМИ

Мы изучили схему «И» с двумя условиями (с двумя иконами Вопрос). Рассмотрим более сложную схему на рис. 64, 65, которая содержит не два, а три условия.

Врач должен получить ответ на три вопроса:

- Чувствует ли беременная женщина движения плода, как обычно?
- В норме ли аускультируемый сердечный ритм плода?
- Прозрачны ли околоплодные воды?

Если на все вопросы получен ответ «Да», это благоприятный исход.

Если же хотя бы на один вопрос ответ «Нет», состояние плода следует оценить, как угрожающее.

Оба алгоритма (рис. 64 и 65) делают одно и то же. Какой из них следует предпочесть? Ответ: левый, так как в нем выполняется эргономичное правило «Чем правее, тем хуже».

Угрожающее состояние плода – это худший вариант. Следовательно, правый алгоритм нарушает требования и бракуется.

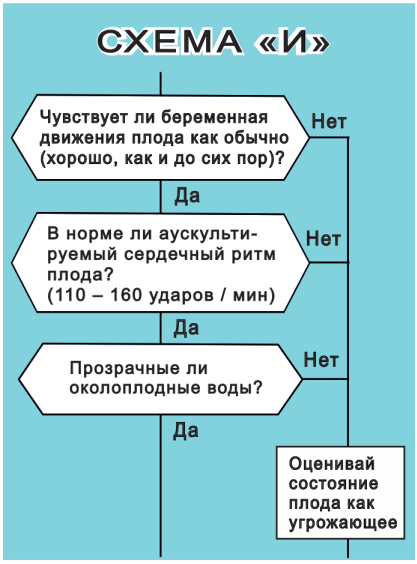


Рис. 64. Схема «И» с тремя условиями. Выход Да *слева*. Картографический принцип соблюдается [233]

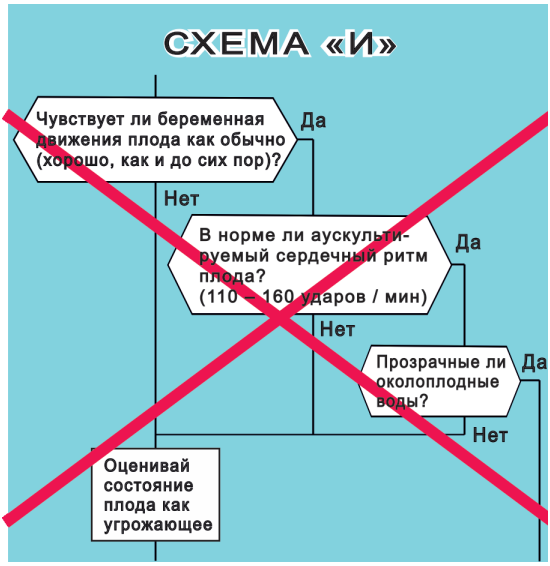


Рис. 65. Схема «И» с тремя условиями. Выход Да *справа*. Картографический принцип нарушен

СХЕМА «И» ДЛЯ ПОЗИТИВНЫХ И НЕГАТИВНЫХ ВОПРОСОВ

Вернемся еще раз к необходимости различать два понятия:

- позитивный вопрос,
- негативный вопрос.

Им соответствуют две разные схемы:

- стандартная схема «И»,
- нестандартная схема «И».

Все четыре примера на рис. 66–69 удовлетворяют правилам ДРАКОНа и являются правильными. Они рекомендуются в качестве образцов для подражания. Схема «И» соответствует выходу Да.

Рис. 66. Стандартная схема «И» с двумя позитивными вопросами [232]

Рис. 67. Нестандартная схема «И» с двумя негативными вопросами [226]

На рис. 66 и 68 (где позитивные вопросы) шампур идет через Да. На рис. 67 и 69 (негативные вопросы) шампур идет через Нет.

МНЕМОНИЧЕСКОЕ ПРАВИЛО

Стандартная схема «И» похожа на «Мачту с парусами», нестандартная – на «Лестницу с уступами».

На рис. 66 мачта имеет два паруса, на рис. 68 – три. Точно так же на рис. 67 лестница имеет два уступа, на рис. 69 – три.

Если добавить еще один или несколько «этажей», мачта будет расти в высоту, а лестница – в ширину.

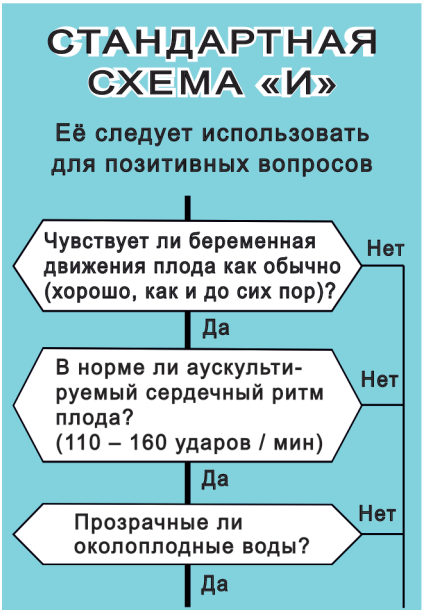


Рис. 68. Стандартная схема «И» с тремя позитивными вопросами [233]

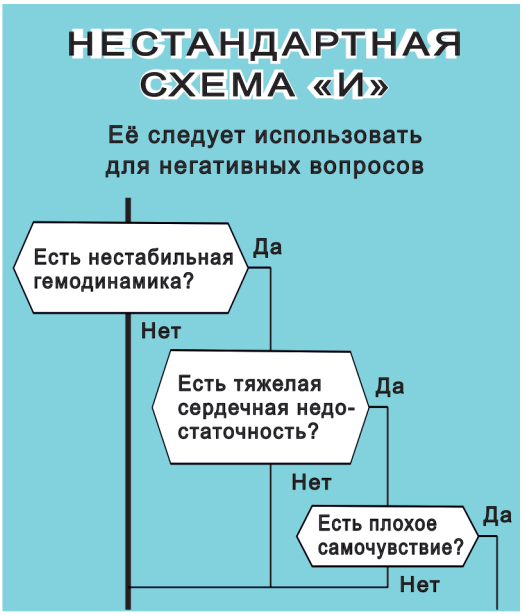


Рис. 69. Нестандартная схема «И» с тремя негативными вопросами [139]

Правило 1 для схемы И

Стандартная схема И (мачта) предназначена для изображения двух и более *позитивных* вопросов.
При этом шампур помечают словом Да.

Правило 2 для схемы И

Нестандартная схема И (лестница) предназначена для изображения двух и более *негативных* вопросов.
При этом шампур помечают словом Нет.

НЕВИДИМАЯ МАТЕМАТИКА. СХЕМА «ИЛИ» И ЗАКОН ДЕ МОРГАНА

Мы рассмотрели свыше тридцати примеров с профессиональными медицинскими текстами, которые выполняют различные логические функции (рис. 36–69). Во всех случаях логические функции реализованы с помощью «невидимой математики», то есть с помощью графики. Графика языка ДРАКОН хороша тем, что позволяет полностью отказаться от логических математических формул.

Как это делается?

Начнем от печки и освежим в памяти рис. 56 и 57. Удалим мысленно медицинский текст в иконах Вопрос. Для краткости заменим его одной буквой (A и B). Результат такого «переодевания» изображен на рис. 70.

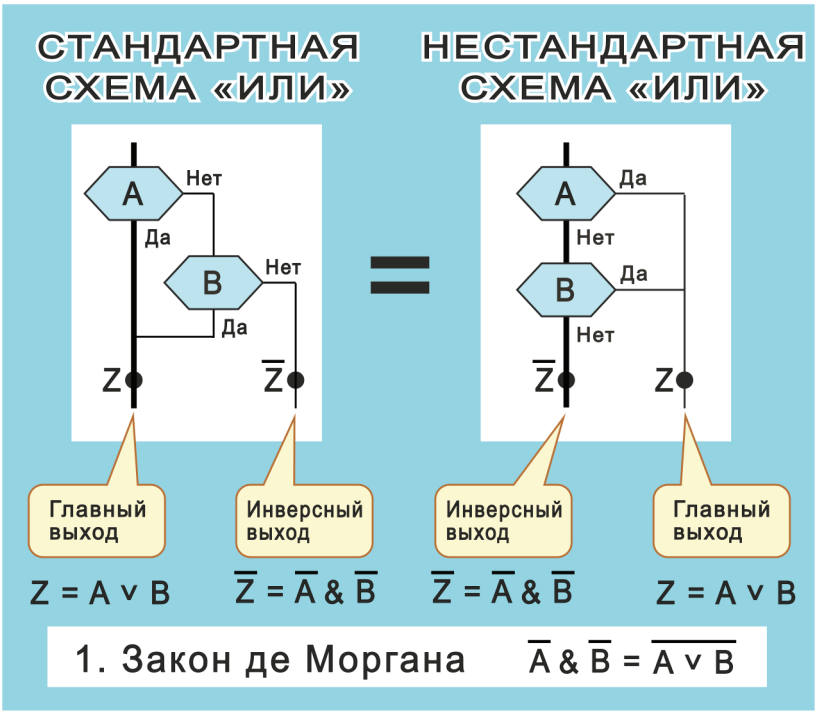


Рис. 70. Две схемы «ИЛИ» в виде лестницы (слева) и мачты (справа) являются равносильными. Они соединены знаком равенства. Связь между главным и инверсным выходами выражает закон де Моргана

Поедем дальше. Главный выход схемы ИЛИ обозначим буквой Z. Второй выход всегда является инверсным и обозначается через $\overline{Z}$, где $\overline{Z}$ – логическое отрицание логической переменной Z.

В нижней части рисунка приведены несложные выкладки, которые показывают, что математическое соотношение между главным и инверсным выходами схемы ИЛИ описывается законом Августа де Моргана (Augustus De Morgan) [140].

Нужны ли эти выкладки для практикующего врача? Конечно, нет. Мы привели их только для того, чтобы проиллюстрировать суть понятия «невидимая математика».

Графика языка ДРАКОН полностью заменяет подобные выкладки, отображая математическую сущность проблемы визуальными средствами.

Вывод состоит в том, что – вместо утомительной работы с логико-математическими формулами и таблицами истинности – врачу достаточно запомнить два мнемонических понятия (Мачта и Лестница) и соответствующие им наглядные зрительные образы.

СХЕМА «И» И ВТОРОЙ ЗАКОН ДЕ МОРГАНА

На рис. 71 та же самая проблема решена для схемы «И».

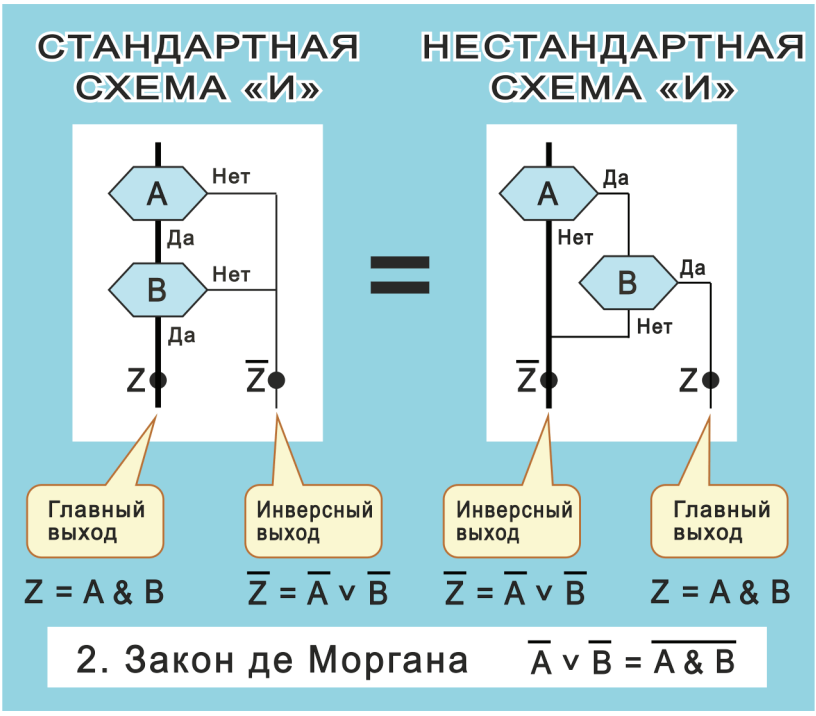


Рис. 71. Две схемы «И» в виде мачты (слева) и лестницы (справа) являются равносильными. Они соединены знаком равенства. Связь между главным и инверсным выходами выражает второй закон де Моргана

Для начала посмотрим на рис. 66 и 67. Медицинский текст в иконах Вопрос на этих рисунках для краткости заменим буквами A и B. Главный выход схемы И обозначим буквой Z. Второй (инверсный) выход является логическим отрицанием Z. Обозначим его через $\overline{Z}$ ($\overline{Z}$ с чертой).

В нижней части рисунка, как и раньше, приведены логические формулы. Отличие в том, что на этот раз они относятся к схеме И.

Обратите внимание. Математическое соотношение между главным и инверсным выходами схемы И описывается не первым, а вторым законом Августа де Моргана.

ЛОГИЧЕСКОЕ ОТРИЦАНИЕ

Предположим, что «В кармане есть деньги». Применим к этой фразе операцию «НЕ» (логическое отрицание). Получим «В кармане денег нет». Идем дальше. Применим операцию «НЕ» к слову «стабильная». Получим «нестабильная».

Усложним задачу. Рассмотрим алгоритм на рис. 72 в левой рамке и применим к вопросу операцию «НЕ» (логическое отрицание). Одновременно переставим слова Да и Нет. Эти два изменения компенсируют друг друга. В итоге образуется точно такой же (равносильный) алгоритм. Он изображен на рис. 72 справа.

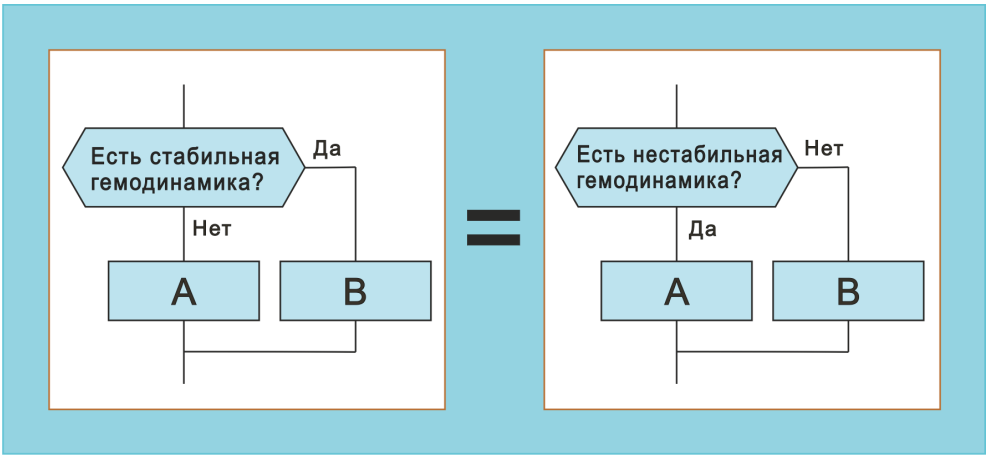


Рис. 72. Пример операции «НЕ» (логическое отрицание). Если в правой развилке удалить отрицательную частицу «не» и поменять местами ответы Да и Нет, получим равносильный алгоритм, нарисованный в левой рамке

Два одинаковых алгоритма, как обычно, соединены знаком равенства.

РОКИРОВКА

Мы приближаемся к самому интересному месту данной главы. Главный секрет связан с понятием *рокировка*.

На рисунках 36–55 мы брали «каждой твари по паре» и попарно анализировали медицинские алгоритмы. Правый алгоритм из каждой пары мы браковали и зачеркивали жирным крестом. Вместо него в качестве образца выбирали левый алгоритм. И так в каждой паре.

Секрет в том, что забракованный алгоритм мы улучшали и превращали в красавчика, используя операцию *рокировка*.

Рокировка обладает удивительным свойством. Она может исправить плохое и создать хорошее. Подобно магическому заклинанию она превра-

щает алгоритмического урода в красавца. Конечно, это возможно лишь тогда, когда оба алгоритма равносильны (рис. 73).

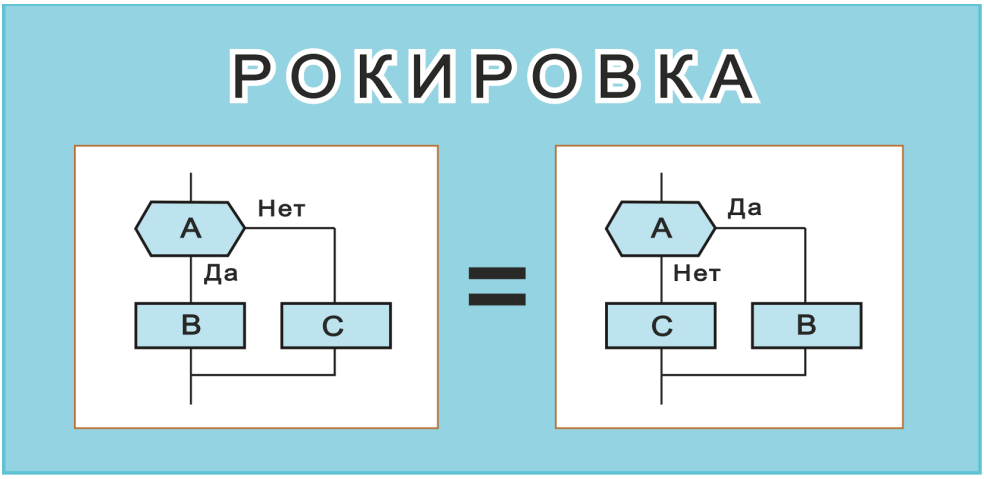


Рис. 73. Рокировка преобразует левую развилку в правую (можно и наоборот). При этом:

- Графика остается неизменной.
- Меняются местами иконы B и C.
- Слова Да и Нет также меняются местами.

Рокировка – операция, которая видоизменяет внешний облик алгоритма, не меняя его по существу. Это значит, что рокировка – равносильное преобразование алгоритма. При рокировке смысл алгоритма не меняется.

Уточним. Операция «рокировка» относится не ко всему алгоритму, а только к одной развилке (как показано на рис. 73).

Говоря упрощенно, при рокировке левая и правая части развилки меняются местами. Назовем эти части *плечами*.

Левое плечо развилки есть маршрут от нижнего выхода иконы Вопрос до точки слияния. Оно содержит ответ «Да», икону B и вертикальную линию (рис. 73, слева).

Правое плечо – путь от правого выхода иконы Вопрос до точки слияния. Оно включает слово «Нет», икону C и соединительные линии.

Что такое рокировка?
Это плеч перестановка!

ПРИМЕР РОКИРОВКИ

Пример показан на рис. 74. В чем заключается рокировка в данном случае? Чтобы ответить, надо сравнить развилки в левой и правой рамке. И посмотреть, что изменилось.

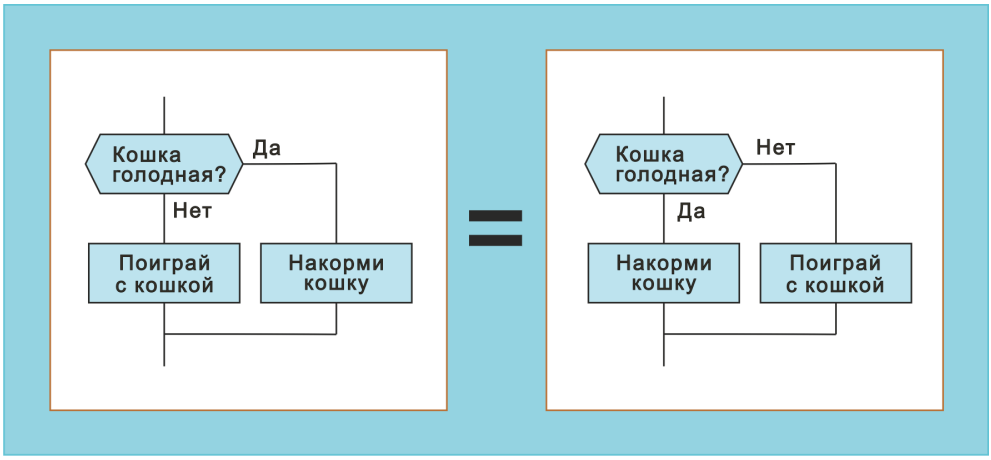


Рис. 74. Пример операции «рокировка». Две схемы являются равносильными. Они соединены знаком равенства. Плохую схему справа можно исправить рокировкой и превратить в хорошую схему слева

Две иконы «Поиграй с кошкой» и «Накорми кошку» поменялись местами. Точно так же переехали слова Да и Нет. Все остальное осталось на прежнем месте.

ЗАЧЕМ НУЖНА РОКИРОВКА

При проведении рокировки следует ответить на два вопроса.

1. Что было и что стало?
2. Зачем нужна рокировка в данном случае?

Предположим, что исходной является схема в правой рамке на рис. 74. Эта схема имеет дефект – нарушено правило «Чем правее, тем хуже». Действительно, голод не тетка. Если кошка голодная, это плохо, если сытая, хорошо.

Недостаток правой схемы заключается в том, что голодная кошка находится на главном маршруте. Это неправильно. Чтобы устранить недочет, надо осуществить рокировку. Рокировка перетащит голодную кошку на боковой маршрут, и все будет в порядке.

После рокировки получим схему в левой рамке на рис. 74. Таким образом, плохую схему в правой рамке можно с помощью рокировки улучшить, облагородить и преобразовать в хорошую схему, находящуюся в левой рамке.

Двинемся дальше. Забудем про голодных кошек и свежим глазом глянем на дружные парочки медицинских алгоритмов на рис. 36–55.

Отметим, что во всех случаях наблюдается одна и та же картина. А именно: в каждой «парочке» исходной является правая, бракованная

схема – та, что зачеркнута. Так вот, эту негодную схему необходимо исправить с помощью спасительной рокировки. И превратить в хороший алгоритм, размещенный в левой части каждой пары.

Это означает, что мы сразу получили десяток убедительных примеров (рис. 36, 38–54). Потому что рокировка позволила исправить дефектную схему и превратить ее в хороший медицинский алгоритм.

РОКИРОВКА МОЖЕТ УЛУЧШИТЬ ЭРГОНОМИЧНОСТЬ АЛГОРИТМОВ

Мы старались показать, что рокировка отличная вещь, праздник души, именины сердца. Осталось выяснить, как с пользой для дела использовать ее на практике. Здесь есть заминка – с непривычки можно попасть в лужу.

Чтобы этого не случилось, рассмотрим довольно сложную задачу. Не одну-единственную развилку, а большой, солидный алгоритм «Обед в ресторане».

На рис. 75 показана плохая дракон-схема. Согласно правилу главный маршрут (жирная линия) должен быть прямым, как стрела, и идти точно по шампуру. А он вместо этого превратился в ломаную-переломаную линию, которая делает невообразимые скачки и путает читателя. Чтобы исправить ошибку, нужно несколько раз сделать рокировку.

Интересно, сколько раз? Оказывается, шесть! Чтобы вылечить больную схему, необходимо выполнить операцию «рокировка» ровно шесть раз!

Начнем по порядку.

Первый раз делаем рокировку в развилке «В меню есть ваш любимый салат?». Во второй – в развилке «Есть хоть какой-то салат?» Двойная рокировка позволяет пустить главный маршрут по шампуру на верхнем участке. Однако внизу главный маршрут по-прежнему совершает неоправданные зигзаги.

Затем делаем рокировку в развилке «Борщ очень вкусный?» И еще раз в развилке «Борщ пересолен?» Тем самым улучшаем среднюю часть схемы.

Нам осталось выпрямить главный маршрут на нижнем участке. Для этого переставляем плечи у двух развилок: «Жаркое как подошва?» и «Другая порция еще хуже?».

В результате шести рокировок наша мечта сбылась! Неэргономичная схема на рис. 75 превратилась в хорошую (эргономичную) схему на рис. 76.

Подведем итоги. Выпрямляя главный маршрут, мы делаем алгоритм более наглядным и легким для понимания. Главный маршрут – путеводная нить алгоритма, позволяющая быстрее уяснить суть дела и не заблудиться в хороводе развилок.

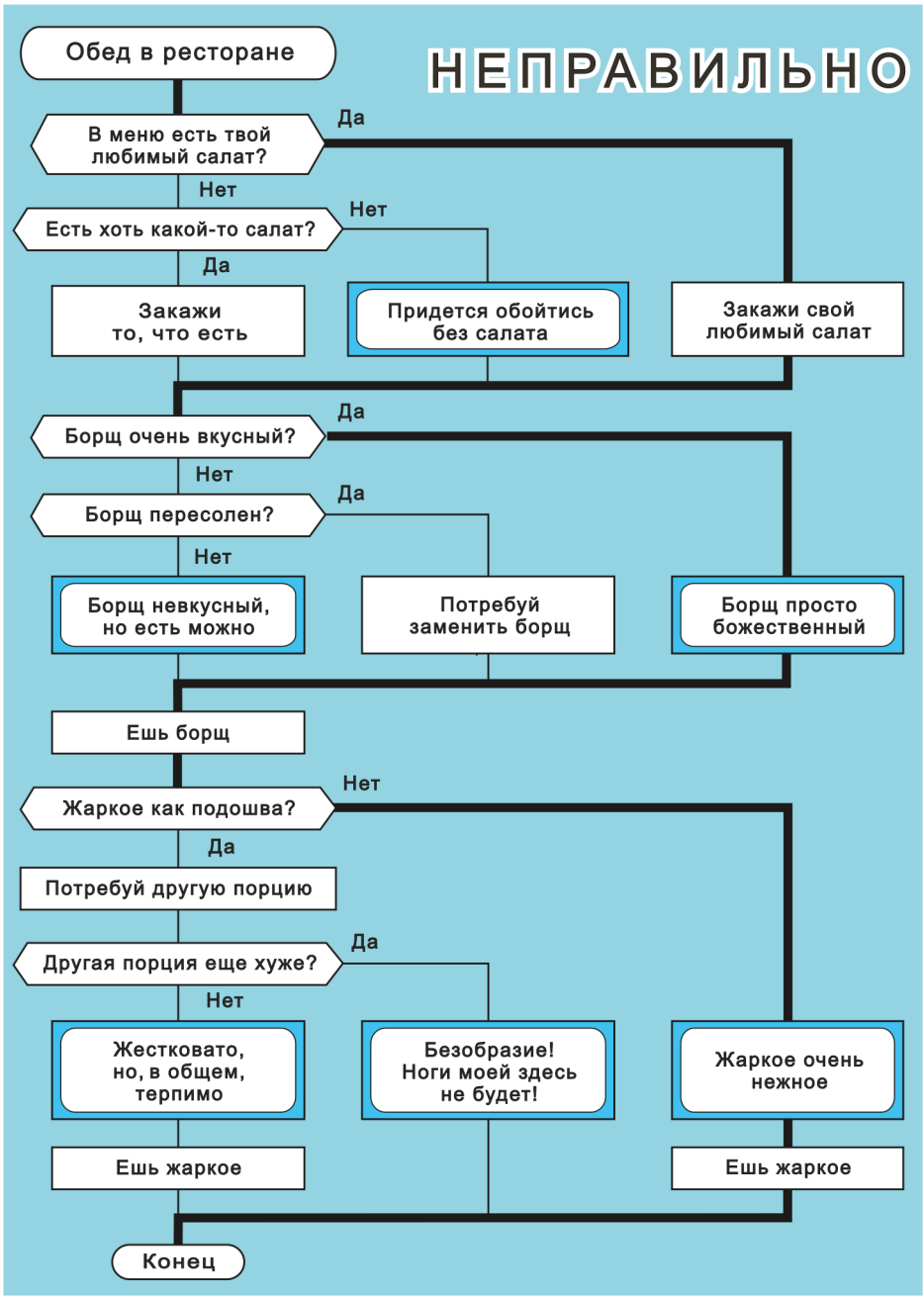


Рис. 75. Плохой дракон-алгоритм. Главный маршрут (жирная линия) все время петляет и делает зигзаги. Его трудно проследить взглядом

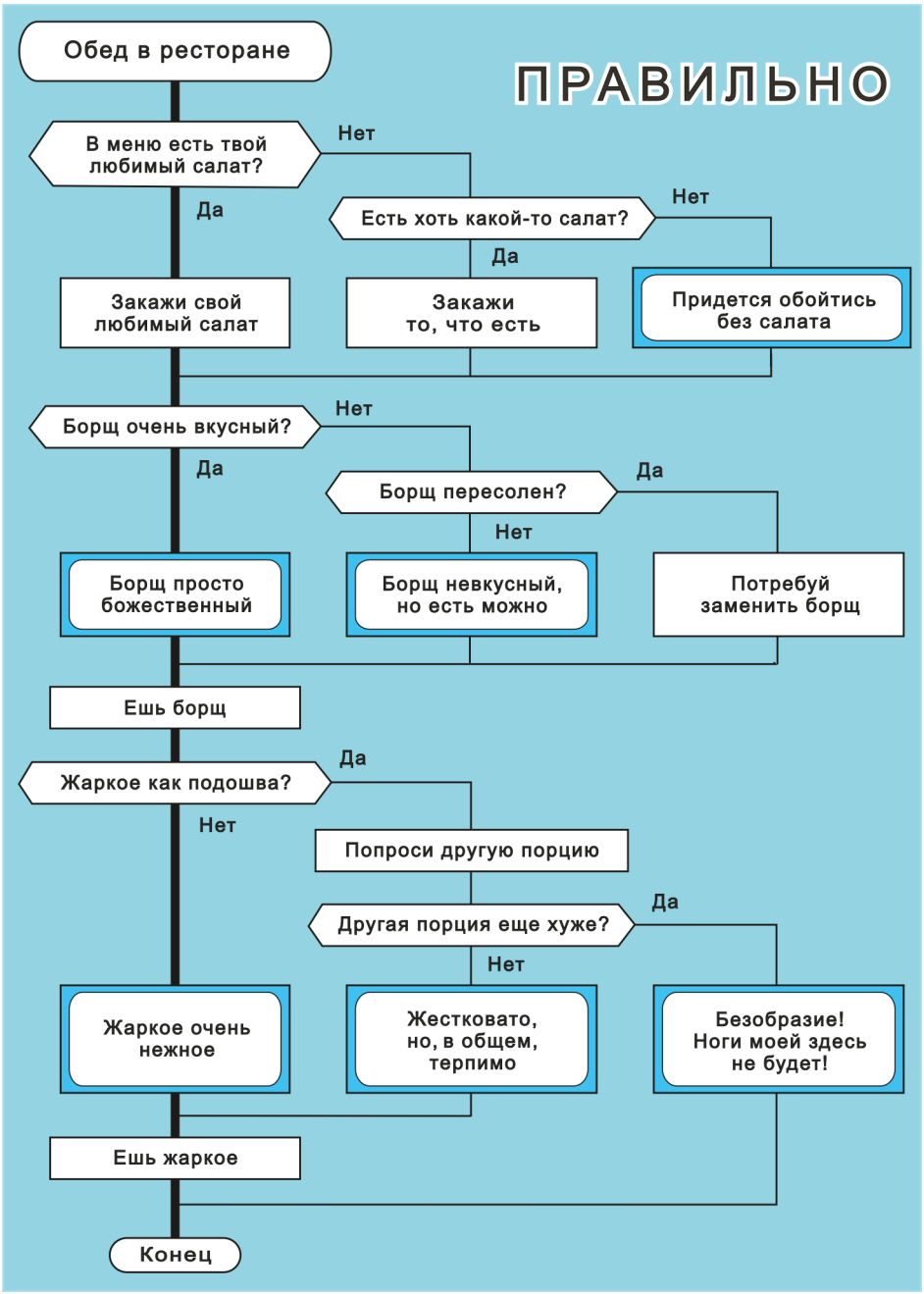


Рис. 76. Хороший дракон-алгоритм. Он получен в результате улучшения схемы на рис. 75. Главный маршрут прямой как стрела. Выполняется правило «Чем правее, тем хуже».

А теперь – самое главное. Мы осуществили выпрямление главного маршрута не случайно, не по принципу «Что хочу, то и ворочу!», а на основании строгого математического закона – *закона рокировки*. Напомним суть закона: рокировка – равносильное преобразование алгоритма. При рокировке смысл алгоритма не меняется.

Полученный результат чрезвычайно важен. Мы воочию убедились, что рокировка позволяет улучшить наглядность и эргономичность алгоритмов.

> Закон рокировки придает нашим эргономическим действиям (позволяющим выпрямить «кривой» главный маршрут) математическую строгость и точность.

ПОПУТНЫЙ СОВЕТ РАЗРАБОТЧИКУ МЕДИЦИНСКИХ АЛГОРИТМОВ

В медицинской литературе процедурные и декларативные знания не разграничены и перемешаны, что вносит серьезные трудности при разработке алгоритмов.

В медицинских текстах и публикациях завершенные алгоритмы в большинстве случаев отсутствуют (такова традиция). Обычно они представлены не целиком, а виде мелких и мельчайших фрагментов (вкраплений), которые переплетаются с другой информацией.

1. Чтобы вычленить алгоритмические знания (вкрапления), их необходимо очистить, отсортировать и отделить от других тем и материалов.
2. Чтобы извлечь медицинский алгоритм из медицинских публикаций, необходимо:
   - Отделить процедурный текст от декларативного.
   - Выявить все условия, в том числе пропущенные в тексте.
   - Выделить все сложные условия, раздробить и заменить каждое из них на несколько простых условий.
   - Обеспечить эргономическую упорядоченность алгоритма, проверив строгое соблюдение картографического принципа.

Картографический принцип обеспечивает порядок, эргономическую наглядность и концептуальное единство во всех частях медицинского алгоритма.

ВЫВОДЫ

1. Язык ДРАКОН позволяет существенно упростить сложную логику медицинских алгоритмов и представить ее в виде наглядных зрительных образов.
2. Вместо утомительной работы с логическими формулами врачу достаточно запомнить два мнемонических понятия (Мачта и Лестница) и соответствующие им зрительные образы.
3. Графические логические схемы подчиняются следующим закономерностям:
    - Стандартная схема ИЛИ (лестница) служит для представления двух и более позитивных вопросов. Шампур помечают словом Да.
    - Нестандартная схема ИЛИ (мачта) предназначена для изображения двух и более негативных вопросов. Шампур маркируют словом Нет.
    - Стандартную схему И (мачту) используют, чтобы показать два и более позитивных вопроса. Возле шампура пишут Да.
    - Нестандартную схему И (лестницу) применяют, чтобы изобразить два и более негативных вопроса. Рядом с шампуром пишут Нет.
4. Графическая схема логического отрицания позволяет изменять и варьировать логические условия в схемах ИЛИ и И, обогащая выразительные возможности графической логики.
5. Рокировка – равносильное преобразование медицинского алгоритма, которое видоизменяет внешний графический облик алгоритма, не меняя его сути. При рокировке смысл алгоритма не меняется.
6. Рокировка позволяет улучшить наглядность и эргономичность медицинских алгоритмов и устранить нарушение картографического принципа.
7. Закон рокировки придает эргономическим действиям разработчика алгоритма (направленным на соблюдение картографического принципа) математическую строгость и точность.
8. В языке ДРАКОН реализован принцип невидимой математики. Это значит, что логические операции: дизъюнкция $\vee$, конъюнкция &, отрицание $\neg$ и законы Августа де Моргана становятся полностью «невидимыми» для врача и реализуются в медицинских алгоритмах с помощью интуитивно понятной графики, что значительно облегчает работу медицинского персонала.

Глава 9

ПОВТОРЕНИЕ МЕДИЦИНСКИХ ДЕЙСТВИЙ, ИЛИ ЦИКЛ

ЧТО ТАКОЕ ЦИКЛ

В медицине часто приходится повторять одни и те же действия. Как изобразить такое повторение? Для этого используют специальный прием, который называется *цикл*.

Цикличные алгоритмы – вещь довольно сложная. Чтобы не затруднять читателя, поясним идею на шутливом сказочном примере. Как говорил великий Блез Паскаль, «предмет математики настолько серьезен, что полезно не упускать случая сделать его немного занимательным» [141].

РАССКАЗ О ЗМЕЕ ГОРЫНЫЧЕ

Предположим, у Змея Горыныча три головы. Чтобы победить Змея, Илья Муромец должен отсечь злодею все головы. То есть три раза исполнить команду «Отруби голову» (рис. 77). Если у Змея пять голов, эту команду придется повторить пять раз (рис. 78).

А если у Змея сто или тысяча голов? Тогда – если действовать в лоб – придется написать подряд сто или тысячу команд. Ясно, что это нелепый путь. Чтобы изложить алгоритм подобным образом, никакой бумаги не хватит!

Чтобы не писать слишком много одинаковых команд, нужно использовать цикл. *Цикл* – это повторяющееся исполнение одних и тех же команд. Фокус в том, что команда в алгоритме записывается *один* раз, а исполняется *много* раз – столько, сколько нужно.

Как работает цикл? Предположим, у Змея только одна голова, которую мы победили с помощью команды 1 (рис. 79). В этом случае на вопрос 2: «У Змея Горыныча остались живые головы?» – отвечаем Нет, и алгоритм заканчивается.

Если же у Змея есть уцелевшие головы, отвечаем Да и, проходя через стрелку, снова попадаем на вход иконы 1. После чего рубим следующую го-

лову. Еще раз задаем вопрос 2. Если у Змея по-прежнему остались головы, отвечаем Да и вновь возвращаемся на вход иконы 1.

Таким образом, исполнение команд 1 и 2 может продолжаться и сто, и двести, и тысячу раз – до тех пор, пока наш алгоритм (управляющий Ильей Муромцем) не отсечет у Змея все головы до последней – см. также рис. 80.

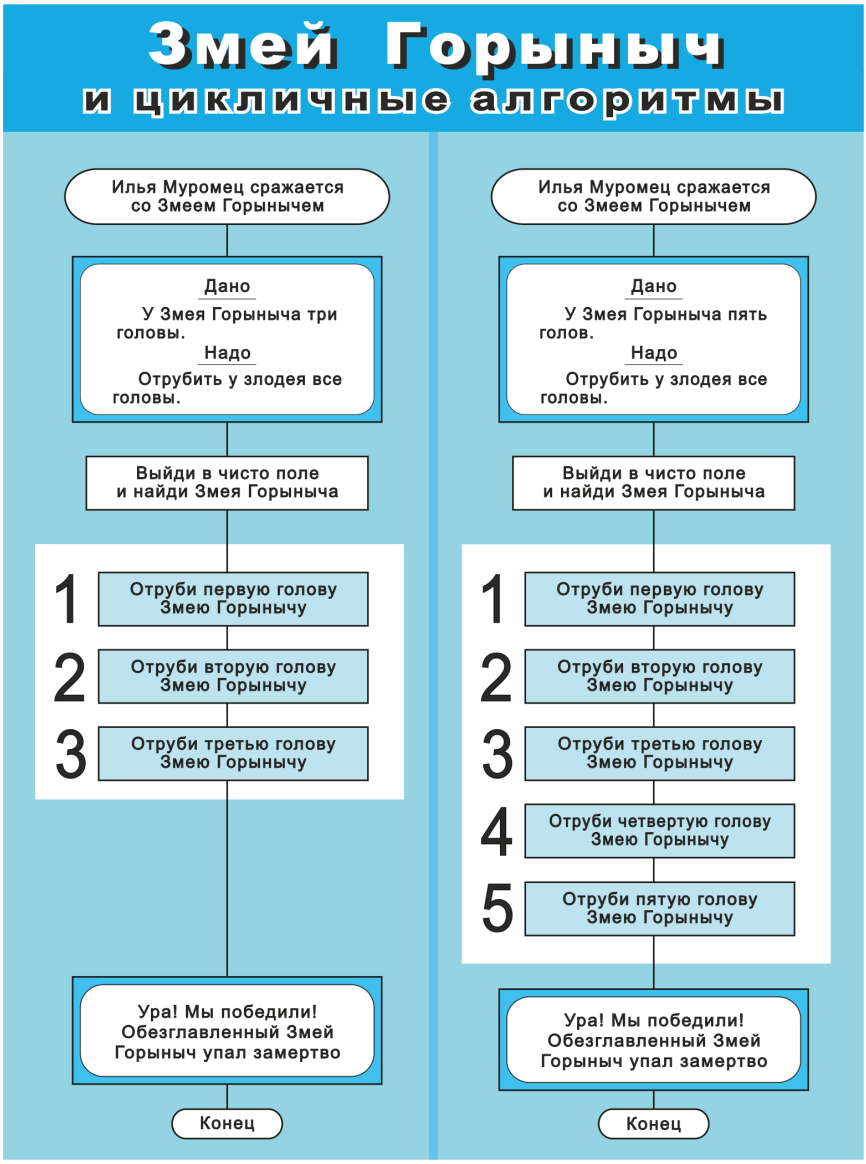


Рис. 77. Плохой (очень длинный) алгоритм. Чтобы отрубить три головы, приходится писать три команды «Отруби голову»

Рис. 78. Плохой (очень длинный) алгоритм. Чтобы отрубить пять голов, надо написать пять команд «Отруби голову». А если у Змея сто голов?

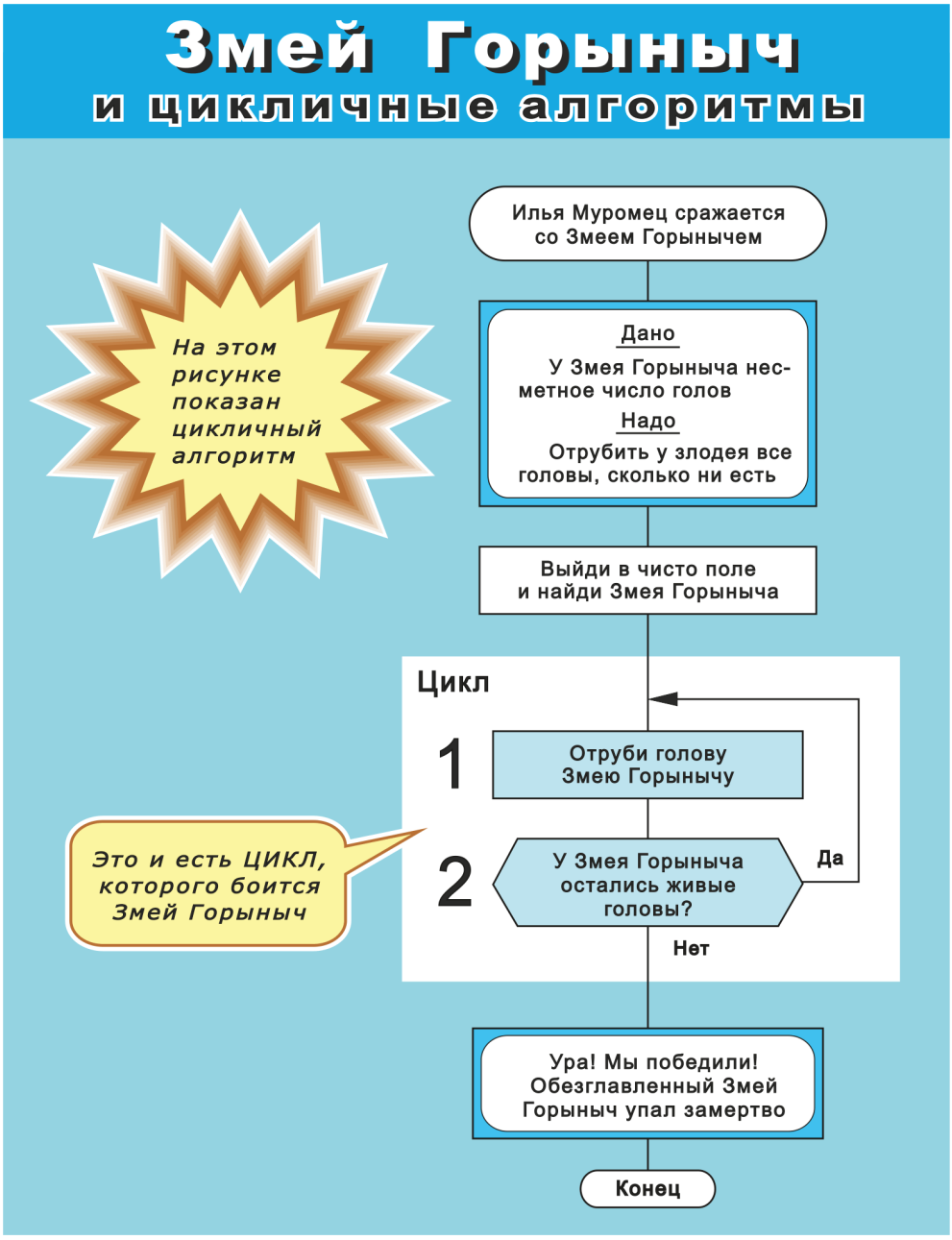


Рис. 79. Хороший (короткий) алгоритм. А почему он короткий? Потому что в нем есть ЦИКЛ. Цикличный алгоритм позволяет отрубить любое число голов. При этом нужна всего одна команда «Отруби голову»

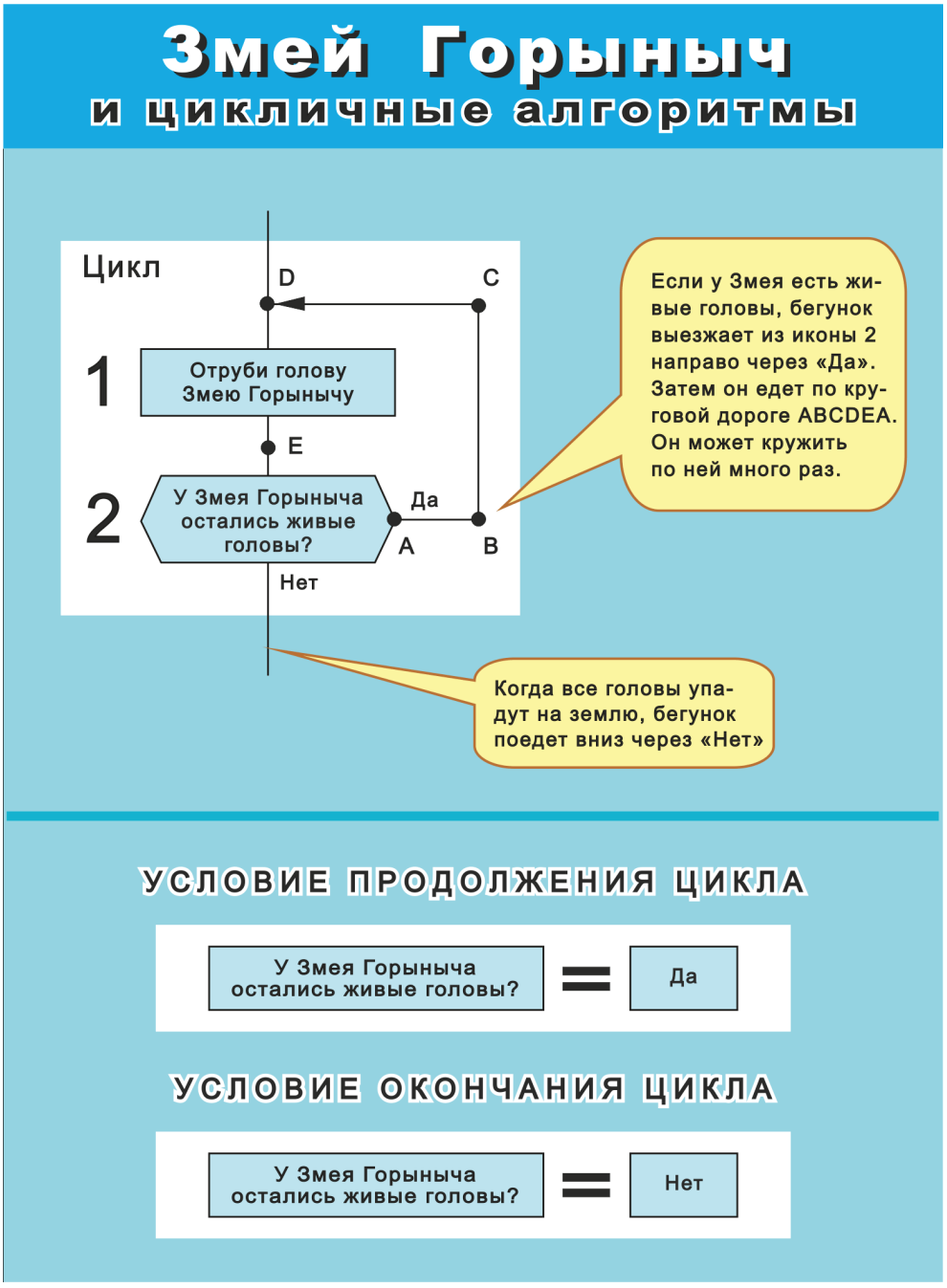


Рис. 80. Цикл – это замечательно! Команды 1 и 2 повторяются много раз, если выполняется УСЛОВИЕ ПРОДОЛЖЕНИЯ ЦИКЛА

УСЛОВИЕ ПРОДОЛЖЕНИЯ И ОКОНЧАНИЯ ЦИКЛА

Следует различать два понятия (рис. 80):

- условие продолжения цикла,
- условие окончания цикла.

Пока выполняется первое условие, действие повторяется снова и снова. В нашем примере это значит, что Илья Муромец безостановочно рубит головы.

Как только исчезает первое условие, сразу же возникает условие окончания цикла:

> У Змея Горыныча остались живые головы? = Нет

Рубить больше нечего. Бегунок на рис. 80 перестает крутиться по круговой дороге и выскальзывает из иконы Вопрос вниз через Нет.

КАК ИЗОБРАЗИТЬ ПОВТОРЕНИЕ ДЕЙСТВИЙ В МЕДИЦИНЕ

При реанимации иногда применяют вдохи «рот в рот». Чтобы выполнить вдох пострадавшему, спасатель должен произвести энергичный выдох.

Предположим, нужно сделать пострадавшему не один искусственный вдох, а много. Как это показать на чертеже? Для этого используют графический цикл.

На рис. 81 в нижней иконе Действие сказано: «Выполни вдох пострадавшему продолжительностью 1 сек.» Выполняя вдох, нужно следить за грудной клеткой пациента. Если она неподвижна (не поднимается), искусственный вдох следует повторять снова и снова. Это легко проследить по алгоритму. Если грудная клетка не шевелится, мы выходим из иконы Вопрос через Нет. А что дальше?

Дальше поворачиваем наверх и по стрелке вновь попадаем в икону «Выполни вдох...». В результате вдох повторяется опять и опять – до тех пор, пока грудная клетка пострадавшего не начнет ритмично подыматься и опускаться.

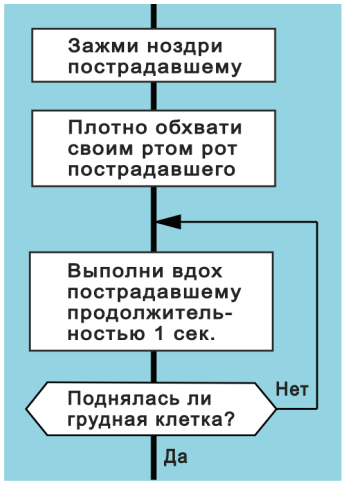


Рис. 81. Применение цикла при реанимации [234]

Таким образом, термин «цикл» используется в медицинских алгоритмах, чтобы обозначить циклическое повторение врачебных действий. На рис. 21 (пункт 3) представлена макроикона «Цикл».

ВЫВОДЫ

1. Если нужно несколько раз повторять медицинскую процедуру, следует использовать графическую фигуру «цикл» (см. макроикону на рис. 21, п. 3).
2. Стрелка цикла всегда закручивается против часовой стрелки.
3. При анализе повторяющихся действий необходимо различать два важных понятия:
    - условие продолжения цикла,
    - условие окончания цикла.
4. Как только будет выполнено условие окончания цикла, он прекращает работу.

Глава 10

СОВМЕСТНАЯ РАБОТА ВРАЧЕЙ

РАБОТА ГРУППЫ ВРАЧЕЙ

До сих пор мы рассматривали одиночную работу врача. Однако в сложных случаях врачи объединяются в бригады и осуществляют медицинскую помощь совместными усилиями.

Отличие в том, что каждый член бригады выполняет свою часть работы не изолированно, а согласованно с другими медиками. Необходима слаженная, синхронная и хорошо скоординированная деятельность персонала. Члены группы работают одновременно и параллельно.

Параллельную работу медицинских работников нужно уметь изображать в виде алгоритма. Подобные алгоритмы называются *параллельными*.

В данной главе представлены параллельные алгоритмы, описывающие работу двух врачей.

СОВМЕСТНАЯ РАБОТА БРИГАДЫ СКОРОЙ ПОМОЩИ

На рис. 82 показана деятельность двух работников скорой помощи. Алгоритм определяет порядок выполнения неотложных действий по спасению пострадавшего мотоциклиста. Он находится без сознания после дорожной аварии с подозрением на перелом позвоночника.

Помощь мотоциклисту оказывают два спасателя. Они работают одновременно и параллельно. На рисунке им присвоены номера 1 и 2. Действия первого спасателя описаны в левом столбце, второго – в правом.

Важно отметить, что действия спасателей скоординированы во времени. Координация показана с помощью двух икон:

- *Начало* синхронной работы спасателей обозначается двойной линией.
- *Конец* синхронного участка изображают одиночной (горизонтальной) линией, в которую сверху вонзаются две стрелки.

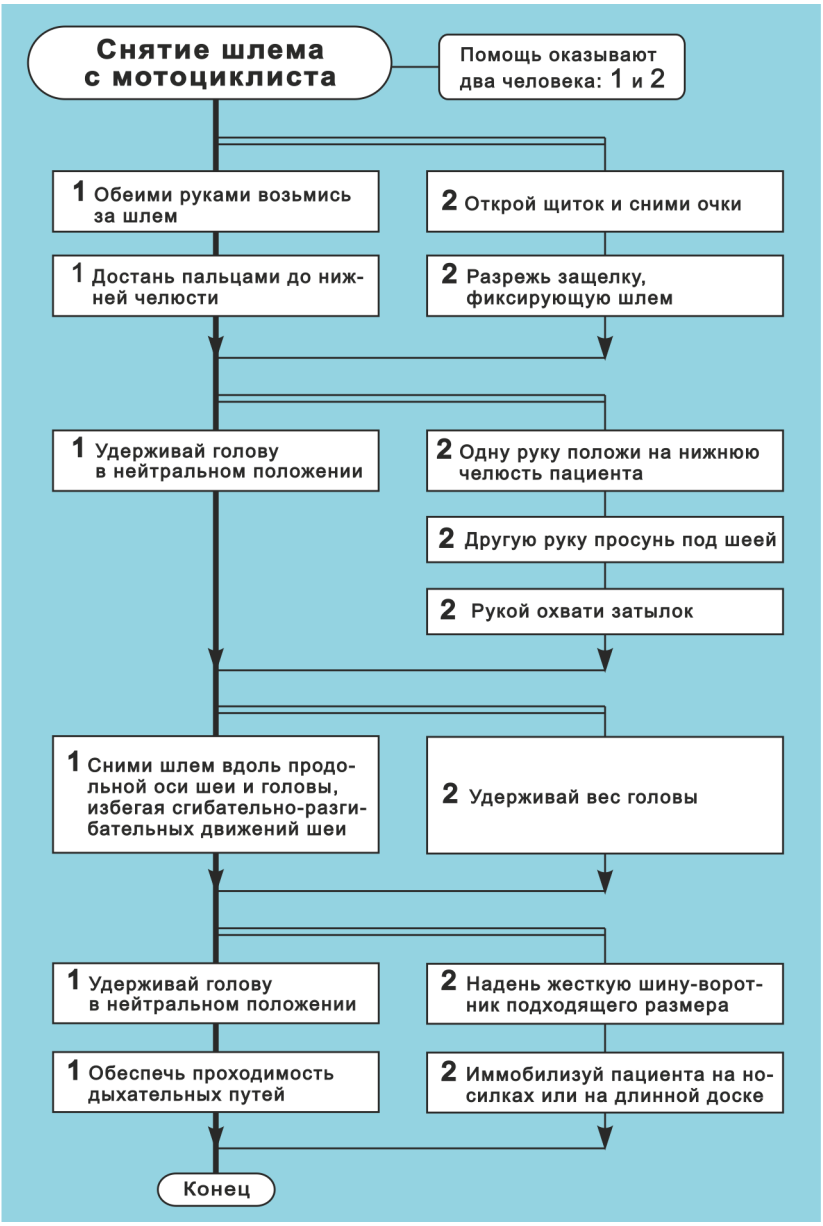


Рис. 82. Алгоритм, описывающий работу двух работников скорой помощи [142]

ПРАВИЛА НУМЕРАЦИИ СПЕЦИАЛИСТОВ

При разработке параллельного медицинского алгоритма необходимо заранее установить четкую нумерацию членов врачебной бригады. Например, для бригады из двух специалистов, необходимо:

- Справа от иконы Заголовок присоединить икону Пояснение с надписью: «Помощь оказывают два человека: **1** и **2**».
- В левой части всех без исключения икон алгоритма поставить номер ответственного за данную операцию члена бригады: **1** или **2**.
- Цифры должны быть крупными и жирными (в пределах разумного), чтобы они сразу бросались в глаза и не вызывали затруднений у читателей.
- Если какую-то операцию медики выполняют вдвоем, в соответствующей иконе слева пишут два номера через запятую: **1, 2**.

Соответствующие иллюстрации показаны на рис. 82 и 83.

Примечание. Икона Пояснение приведена в справочнике на рис. 20, пункт 19.

ДВУХПОТОЧНЫЙ УЧАСТОК

Двухпоточный участок – часть параллельного алгоритма, обладающая следующими свойствами:

- Начало участка обозначено двойной линией, а конец – одиночной.
- В начале участка единый маршрут (поток) раздваивается и превращается в два: левый и правый. В конце участка, наоборот, два потока объединяются и превращаются в один.
- Двухпоточный участок решает строго определенную функциональную задачу, требующую синхронных действий членов медицинской бригады.

Сверхзадача алгоритма на рис. 82 соответствует принципу «Не навреди». Любое небрежное или неосторожное действие спасателей может усугубить состояние мотоциклиста и нанести ему непоправимый вред. Исходя из сверхзадачи, алгоритм разбит на четыре двухпоточных участка, каждый из которых выполняет свою частную задачу.

- *1-й участок.* Выполнить подготовительные операции: открыть щиток, снять очки, разрезать защелку.
- *2-й участок.* Установить руки второго спасателя в нужное положение, чтобы подготовиться 3-му участку (где надо удерживать вес головы).
- *3-й участок.* Снять шлем мотоциклиста.
- *4-й участок.* Надеть жесткую шину-воротник и иммобилизовать пострадавшего на носилках или доске.

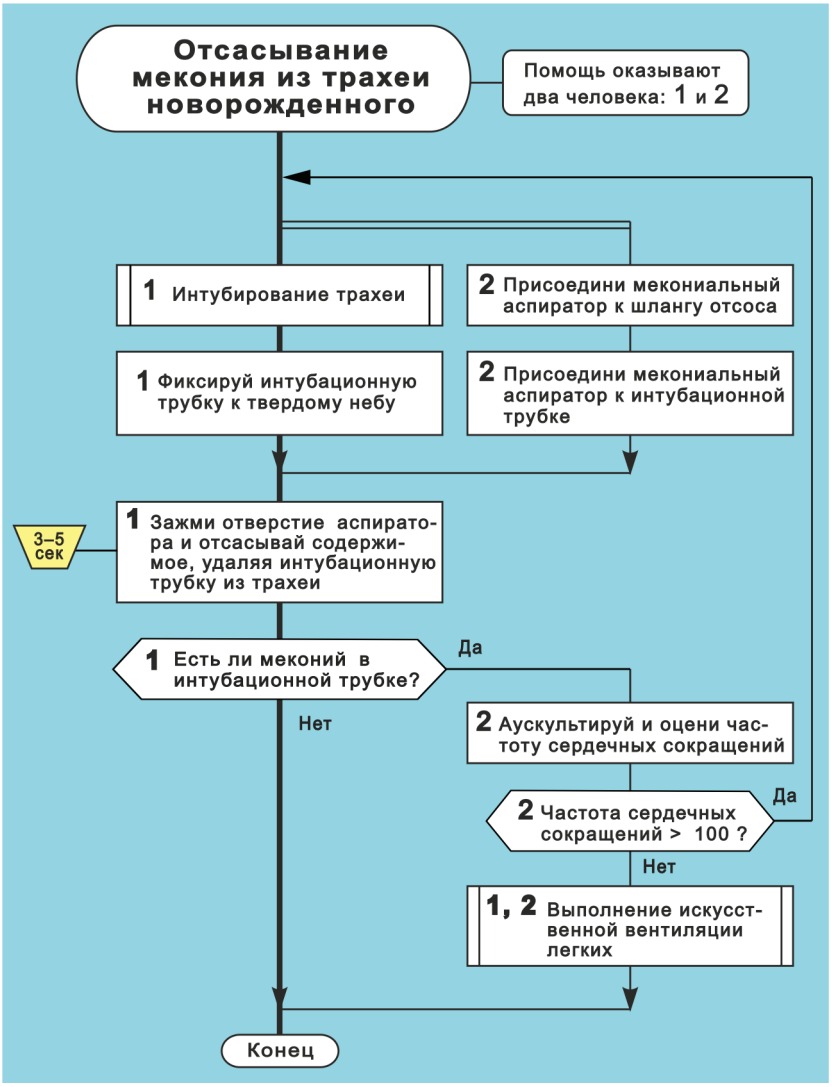


Рис. 83. Алгоритм «Отсасывание мекония из трахеи новорожденного» [143]

СОВМЕСТНАЯ РАБОТА ПРИ ОТСАСЫВАНИИ МЕКОНИЯ ИЗ ТРАХЕИ НОВОРОЖДЕННОГО

Рассмотрим более сложный случай. В алгоритме на рис. 83 применяются три интересных приема:

- совместная работа врачей,
- цикл,
- икона Вставка.

Алгоритм выполняют два врача, которым на чертеже присвоены номера **1** и **2**. Действия первого врача описаны в левом столбце, второго – в правом.

Действия врачей согласованы во времени. Начало совместной работы показано в виде двойной линии, конец – одиночной.

Тонкость состоит в том, что бегунок в верхней части чертежа (ниже двойной линии) раздваивается и одновременно проходит по двум маршрутам: левому и правому. Затем (в конце двухпоточного участка) два бегунка сливаются в один. Это происходит выше иконы «Зажми отверстие аспиратора...». Далее следует один-единственный бегунок.

Тем самым бегунок наглядно демонстрирует групповой характер работы врачей.

Обратите внимание. В данном алгоритме совместная работа отражается двумя способами, именно:

- двухпоточный участок.
- нумерация специалистов,

В верхней части рисунка используются оба способа, а в нижней только один – нумерация специалистов.

Перейдем к анализу цикла. Как его обнаружить?

Найдите горизонтальную стрелку (именно горизонтальную). Затем пройдите по длинной линии, исходящей из иконы Вопрос «Частота сердечных сокращений >100?» через Да.

Проследите карандашом путь, идущий после стрелки вниз через иконы:

- «Интубирование трахеи».
- «Фиксируй интубационную трубку...».
- «Зажми отверстие аспиратора...».
- «Есть ли меконий в интубационной трубке?» Да.
- «Аускультируй и оцени частоту...».
- «Частота сердечных сокращений >100?» Да.

Если на оба вопроса постоянно дается ответ Да, ваш карандаш будет двигаться по кругу против часовой стрелки, описывая все новые и новые круги и никогда не остановится. Получится так называемый бесконечный цикл, из которого нет выхода.

В реальности такого быть не может по двум причинам.

В благополучном случае весь меконий из трахеи будет отсосан, и мы выйдем из иконы Вопрос «Есть ли меконий в интубационной трубке?» через Да.

В неблагоприятном случае (когда частота сердечных сокращений новорожденного менее 100 ударов в минуту, то есть новорожденный неактивный), следует незамедлительно начать искусственную вентиляцию легких. На алгоритме хорошо видно, что из иконы Вопрос «Частота сердечных сокращений >100?» мы выходим через Да.

Вставка. На рис. 83 используются две иконы Вставка:

- «Интубирование трахеи»,
- «Выполнение искусственной вентиляции легких».

Каждая из них представляет собой сложный алгоритм, который обязательно должен быть раскрыт где-нибудь в подходящем для этого месте. Причем читатель должен иметь возможность попасть в это место и ознакомиться с алгоритмом. Если такой возможности нет, значит Вставка превращается в шараду: «Пойди туда, не знаю куда».

В таких случаях говорят, что медицинский алгоритм выполнен некачественно и раскрыт не полностью. Это грубая алгоритмическая ошибка.

Примечание. Описание иконы Вставка дано также в главе 7 на рис. 23 и 24.

ИКОНА «ВРЕМЯ»

Давайте посмотрим на рисунок 83 через увеличительное стекло и найдем желтую икону Время. Чтобы она всегда была под рукой, перенесем ее вместе с соседкой на рис. 84.

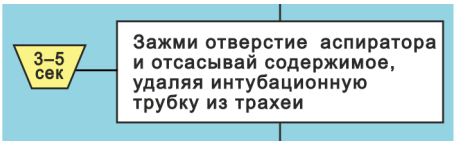


Рис. 84. Икона Время

Мы видим, что икона Время всегда образует пару. Она прицеплена слева к своей хозяйке – иконе Действие и указывает *длительность выполнения операции*.

Но это не все. Икона Время выполняет более тонкую и важную функцию. Поскольку речь идет о спасении жизни новорожденного, она подсказывает врачу, что он обязан выполнить действие БЫСТРО, за 3–5 секунд, потому что при реанимации счет идет на секунды.

Конечно, можно поступить по-другому и обойтись без иконы Время. Для этого нужно:

- перетащить информацию о длительности в «хозяйскую» икону Действие;
- в последней написать: «Зажми отверстие аспиратора и отсасывай содержимое, удаляя интубационную трубку из трахеи. Длительность этого процесса равна 3–5 секунд». Этот вариант показан на рис. 85.

Однако такой способ намного хуже. Беда в том, что высказывание получилась слишком длинное и неудобное для чтения. Чтобы облегчить труд читателя, длинную словесную глыбу лучше разбить на две части. И поместить их в две разные иконы – Время и Действие. Именно так сделано на рис. 84.

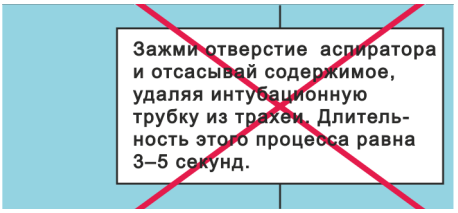


Рис. 85. Вариант без иконы Время

При этом необходимость читать длинный и громоздкий текст отпадает. Икона Время быстро и элегантно предоставляет врачу важную информацию, избавляя его от необходимости выискивать предупреждение о времени в темных закоулках мудреного словесного абзаца.

Таким образом, икона Время позволяет качественно улучшить алгоритм. Она создает удобное средство для синхронизации и координации временных соотношений в медицинских алгоритмах.

Чтобы медицинская наука интенсивно развивалась, она должна овладеть новым языком для описания медицинских знаний. Этот язык должен быть очень хорошим. Он должен позволять создавать высококачественные медицинские алгоритмы – доступные, удобные, легкие для понимания. Язык ДРАКОН призван способствовать достижению этой цели. На отдельных частных примерах мы стремимся показать различные возможности языка.

ВЫВОДЫ

1. В сложных случаях врачи объединяются в бригады и осуществляют медицинскую помощь коллективно.
2. Параллельные медицинские алгоритмы служат для описания синхронной, точно скоординированной работы членов медицинской бригады.
3. При разработке параллельного медицинского алгоритма используют:
   - правила нумерации специалистов,
   - правила двухпоточного участка.
4. Начало двухпоточного участка обозначают двойной линией, конец – одиночной.
5. Параллельный алгоритм может содержать несколько синхронных (двухпоточных) участков, соединенных последовательно.
6. Координацию длительности операций удобно задавать с помощью иконы Время.

Глава 11

НОВЫЙ СИЛУЭТ МЕДИЦИНСКОГО АЛГОРИТМА

ПРИМИТИВ И СИЛУЭТ

Язык ДРАКОН имеет две алгоритмические конструкции:

- *примитив* (для описания простых медицинских алгоритмов),
- *силуэт* (для сложных и сверхсложных).

В предыдущих главах мы рассматривали простейшие случаи и использовали конструкцию «примитив». Примеры примитивов представлены на рис. 1–4, 23–26, 28, 75–79, 82, 83.

На рис. 8, 9, 14, 15, 29–34, 36–75, 80, 81, 84, 85 показаны фрагменты алгоритмов, которые можно применять где угодно – и в примитиве, и в силуэте.

Чем отличается примитив от фрагмента? Примитив – простой, но полностью законченный алгоритм, имеющий иконы Заголовок и Конец. Фрагмент – недоделанная часть алгоритма, в ней нет ни Заголовка, ни Конца.

ПРИНЦИПИАЛЬНЫЙ НЕДОСТАТОК ПРИМИТИВА

Важно помнить, что примитив имеет ограниченные возможности. Во многих случаях (за исключением самых простых) он не позволяет изобразить ЛЮБОЙ медицинский алгоритм.

От этого недочета свободен силуэт, который можно применять во всех без исключения случаях, в том числе в наиболее сложных и изощренных. Однако для самых простых случаев все-таки выгоднее использовать примитив.

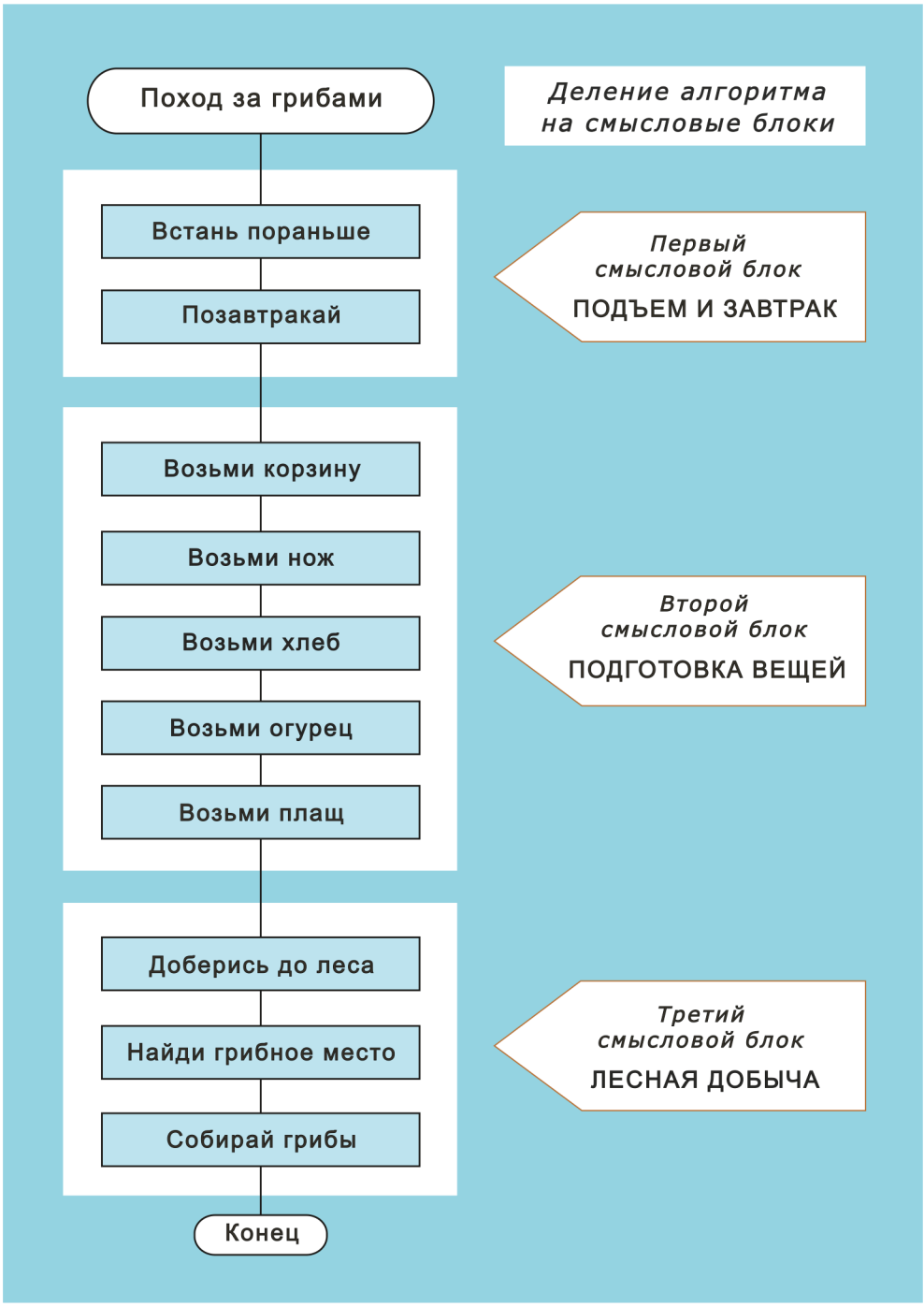


Рис. 86. Алгоритм «Поход за грибами»

ЧТО ТАКОЕ СИЛУЭТ

Силуэт – чрезвычайно мощная алгоритмическая конструкция, обладающая большими выразительными возможностями. Она позволяет описывать медицинские алгоритмы любой сложности, причем делать это в привлекательной для врачей форме.

Силуэт является основным достоинством языка ДРАКОН, его главным оружием.

Основным элементом силуэта является «ветка». С нее-то мы и начнем.

ВЕТКА

На рис. 86 представлен алгоритм «Поход за грибами», содержащий довольно большую последовательность действий. Иной раз такая последовательность может оказаться чрезмерно длинной и утомительной для чтения.

Можно ли облегчить восприятие и анализ подобных «длиннющих» алгоритмов? Можно ли сделать «долговязый» алгоритм обозримым и удобным для быстрого понимания?

Да, можно. Чтобы облегчить работу читателя и сделать алгоритм эргономичным, надо заблаговременно разрезать «длинную кишку» и разбить ее на смысловые части. Сделать это нетрудно. «Поход за грибами» (рис. 86) – это алгоритмический рассказ, в котором можно выделить три крупных куска, три самостоятельных темы:

- Подъем и завтрак.
- Подготовка вещей.
- Лесная добыча.

Каждую тему можно нарисовать в виде ветки. Результат изображен в виде силуэта на рис. 87.

Названия двух новых икон приведены в таблице.

Икона	Название иконы	Пояснение
	Имя ветки	Обозначает начало ветки
	Адрес	Обозначает конец любой ветки, кроме последней

Вопрос. Где начало и конец у первой ветки на рис. 87?

Ответ. Начало – икона «Подъем и завтрак». Конец – «Подготовка вещей». Между началом и концом размещается тело ветки. Оно содержит команды «Встань пораньше» и «Позавтракай».

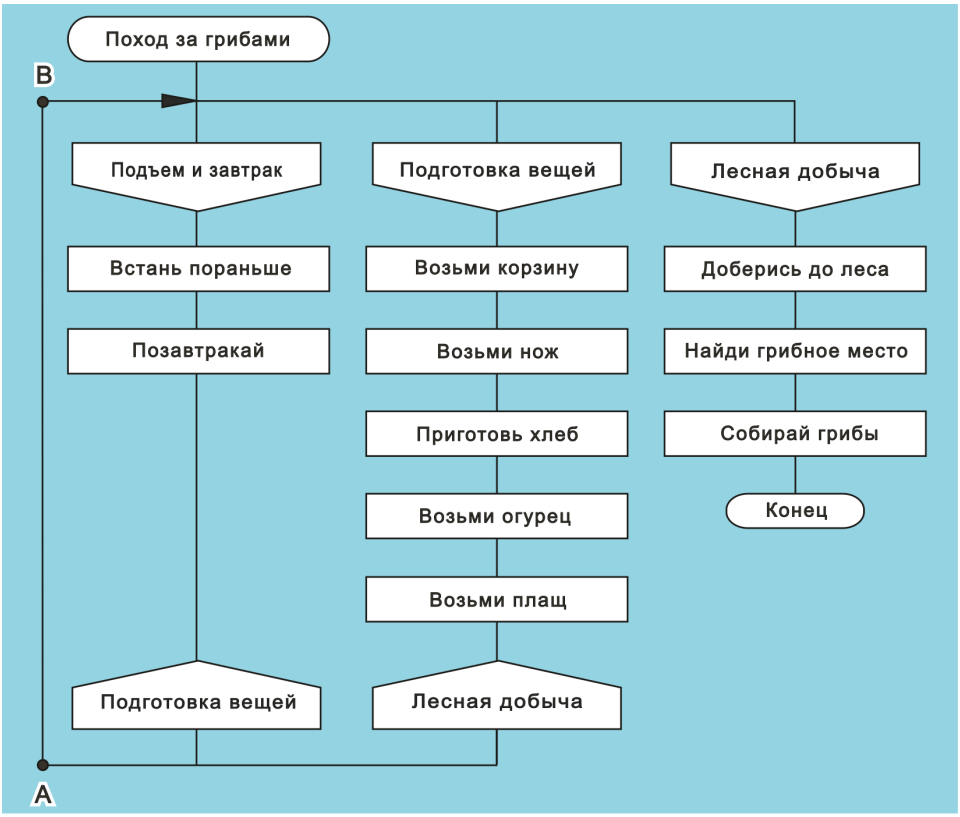


Рис. 87. Алгоритм силуэт «Поход за грибами». Сравни с рис. 86

Чтобы разделить проблему на несколько смысловых частей (например, на пять), разбейте алгоритм на пять веток.

Зачем нужна ветка?

Чтобы помочь врачу разбить медицинский алгоритм на смысловые части. И, что очень важно, дать частям удобные и точные названия. Название каждой части пишут в иконе «Имя ветки».

Что такое ветка?

Это смысловая часть алгоритма, которая содержит:

- Икону «Имя ветки» (в ней пишут название смысловой части).
- Тело ветки, состоящее из команд, записанных в иконах.
- Одну или несколько икон Адрес (в любой ветке, кроме последней).

ШПАРГАЛКА

Икона Имя ветки очень похожа на икону Вариант. С непривычки их можно перепутать. Чтобы избежать досадных ошибок, сравните иконы на рис. 88 и 89.

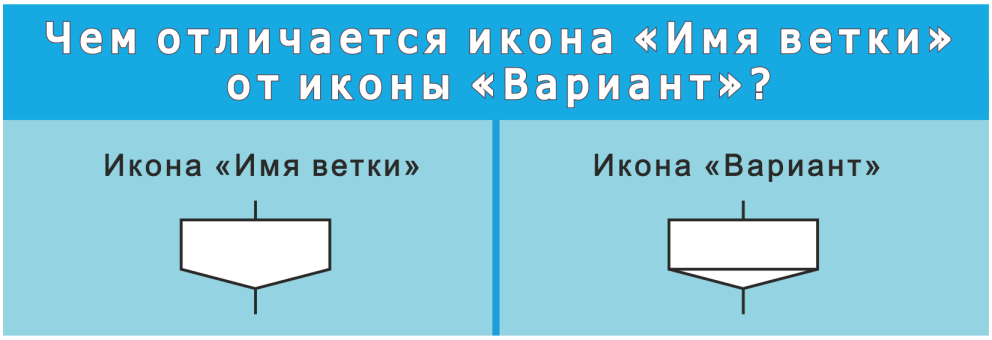


Рис. 88. У иконы «Имя ветки» одна горизонтальная линия

Рис. 89. У иконы «Вариант» две горизонтальные линии

КАК ЧИТАТЬ СИЛУЭТ

Сначала читают Заголовок (рис. 87), затем по шампуру – крайнюю левую ветку. В самом низу находится икона Адрес. Зачем она нужна?

В ней пишут Имя следующей (по порядку исполнения ветки. Икона Адрес не позволит вам сбиться с пути. Она подсказывает, какую ветку следует читать дальше.

Вход в ветку возможен только через ее начало. Выход из последней ветки осуществляется через икону Конец.

ШАПКА

Многие алгоритмы очень сложны. Чтобы разобраться в тонкостях такого алгоритма, нужно прилагать большие усилия и тратить много времени.

Подобную практику следует признать порочной. Алгоритмы можно и нужно сделать легкими для восприятия (эргономичными). Для этого надо давать читателю маленькие, но умные подсказки, проглотив которые, он мог бы легко сориентироваться в задаче и быстро понять, что к чему. Одной из таких подсказок

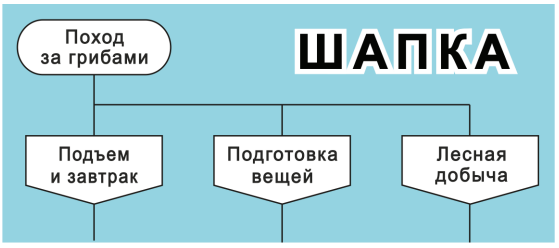


Рис. 90. Шапка – важный элемент силуэта. Она показывает структуру силуэта

служит *шапка*. Название объясняется тем, что шапка находится вверху, «на голове» у алгоритма.

Назначение шапки – помочь читателю мгновенно (за несколько секунд) сориентироваться в задаче и, расчленив ее на части, увидеть структуру алгоритма (рис. 90). Причем увидеть не в фигуральном смысле слова, не с помощью воображения, не духовным оком, а своими двумя глазами – на бумаге или экране.

Что такое шапка?

Это верхняя часть силуэта, которая включает Заголовок алгоритма и комплект икон «Имя ветки».

ТРИ ЦАРСКИХ ВОПРОСА

Сталкиваясь с новой незнакомой задачей, мы обычно желаем получить ответ на три царских (наиболее важных) вопроса:

1. Как называется задача?
2. Из скольких частей она состоит?
3. Как называется каждая часть?

Именно с этих вопросов начинается наше знакомство с задачей при рациональном подходе к делу.

Эргономическая хитрость состоит в том, что шапка, угадывая тайное желание читателя, дает ему подсказку – ответ на все царские вопросы.

Вот ответы для рис. 87.

- Как называется задача? (*Читаем Заголовок алгоритма*).
  Поход за грибами.
- Из скольких частей она состоит? (*Считаем иконы «Имя ветки»*).
  Из трех.
- Как называется каждая часть? (*Читаем текст в иконах «Имя ветки»*).
  1. Подъем и завтрак.
  2. Подготовка вещей.
  3. Лесная добыча.

КАК БЕГУНОК ДВИЖЕТСЯ ПО СИЛУЭТУ

Вспомним, что бегунок – воображаемая точка, которая поочередно пробегает все иконы одного из маршрутов, перемещаясь из начала в конец. Как он ведет себя в силуэте?

Как обычно – мчится по алгоритмической дорожке от иконы Заголовок до иконы Конец. Выехав из Заголовка бегунок скользит вниз по крайней левой ветке. Он движется через станции (рис. 87):

- Подъем и завтрак.
- Встань пораньше.
- Позавтракай.
- Подготовка вещей.

Икона Адрес – последняя станция первой ветки. Куда ехать дальше? Ответ содержится внутри самой иконы. Эта икона потому и зовется «Адрес», что сообщает адрес (название) следующей станции. В данном случае она говорит: следующая станция называется «Подготовка вещей».

Из рис. 87 видно, что данная станция находится в начале второй ветки. Но как бегунок доберется туда? По какой линии?

Ответ прост. Выехав из иконы Адрес, бегунок сворачивает налево и попадает в точку *A* (рис. 87). Потом движется вверх к точке *B*. Затем едет направо по стрелке и въезжает в верхнюю икону «Подготовка вещей».

Дальше все происходит аналогично. Бегунок скользит вниз по второй ветке. Добравшись до иконы Адрес, узнает адрес следующей станции («Лесная добыча»). Затем огибает схему по линии *AB*, попадает в начало третьей ветки и спускается по ней до конца. На этом выполнение алгоритма заканчивается.

В ЧЕМ СЕКРЕТ ИКОНЫ «АДРЕС»

Сейчас мы поступим, как чеховский злоумышленник – будем разбирать рельсы. Имеются в виду линии, окаймляющие алгоритм на рис. 87. Сотрем линию *AB*. Уберем также горизонтальные линии (шины), проходящие через точки *A* и *B*. Результат представлен на рис. 91.

Как будет работать силуэт после этих исправлений?

К нашему удивлению, отсутствие рельсов никак не сказывается на работе алгоритма. В этом легко убедиться. Выехав из иконы Заголовок, бегунок движется вниз по крайней левой ветке. Опустившись до конца ветки, бегунок попадает в икону Адрес.

Казалось бы, это тупик. Рельсы кончились, дальше ехать некуда. Но это не так. Ведь в иконе Адрес записан адрес следующей станции («Подготовка вещей»). Зная адрес, бегунок прыгнет туда, куда нужно – в начало второй ветки. И поедет вниз. Добравшись до конца второй ветки, он совершит второй прыжок. И попадет в третью ветку. И так далее.

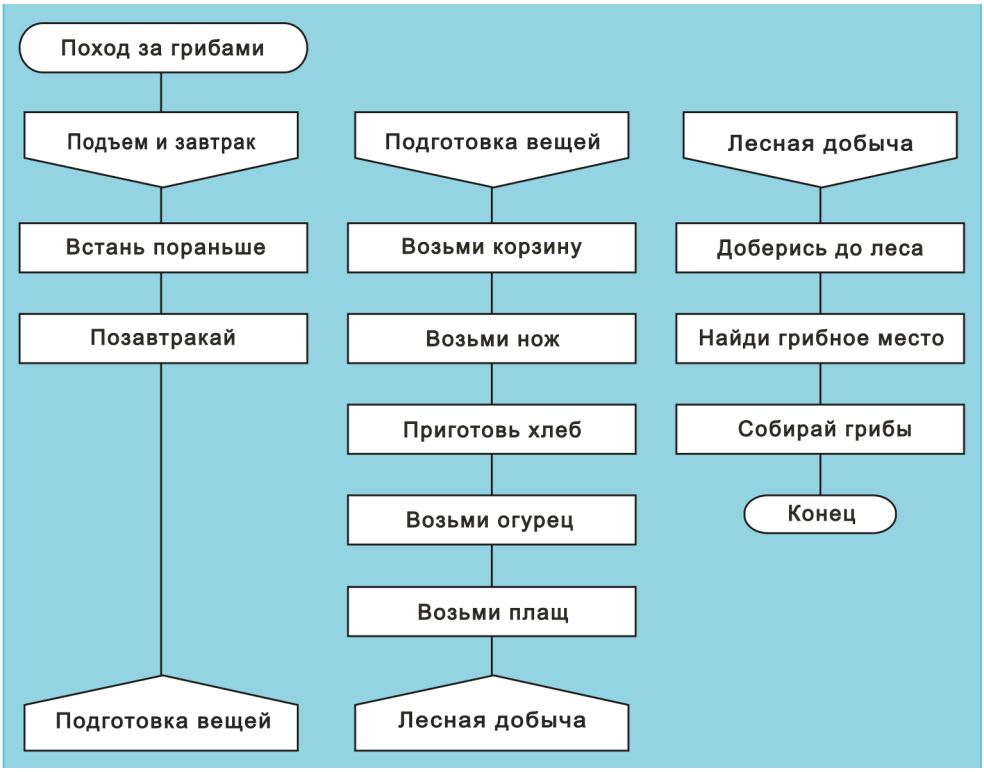
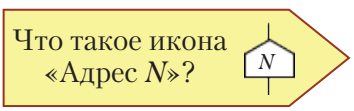


Рис. 91. Схема «с разобранными рельсами». Сравни с рис. 87

Таким образом, икона «Адрес *N*» – это команда «Прыгни в начало ветки *N*». Проще говоря, данная команда передает приказ: «Брось данную ветку и начни выполнять ветку *N*». (Выполнять ветку – значит исполнять записанные в ней команды).



Это команда, заставляющая бегунок прыгнуть в начало ветки *N*.

Переходя от рис. 87 к рис. 91, мы для упрощения стёрли несколько линий. И сразу убедились, что для работы алгоритма они совершенно не нужны. Маршрут бегунка определяют не они, а указания, записанные в иконе Адрес.

Тем не менее, указанные линии не следует удалять по эргономическим соображениям. Дело в том, что обрамляющие линии зрительно «склеивают» разрозненные куски алгоритма. Они превращают их в приятный для глаза целостный зрительно-смысловой образ. И наоборот, устранение скрепляющих линий приводит к тому, что схема зрительно рассыпается на части, что сбивает с толку читателя (рис. 91).

ВХОД И ВЫХОДЫ ВЕТКИ

Ветка имеет один вход. И один или несколько выходов. На рис. 87 все ветки имеют только один выход. На рис. 92 ситуация иная – ветка «Подъем и завтрак» обладает двумя выходами:

- Идем по грибы.
- Идем за ягодами.

Это позволяет с помощью переключателя «Что нужно?» организовать разветвление и перейти в две различные ветки. Наличие у ветки нескольких выходов позволяет расширить выразительные возможности языка ДРАКОН.

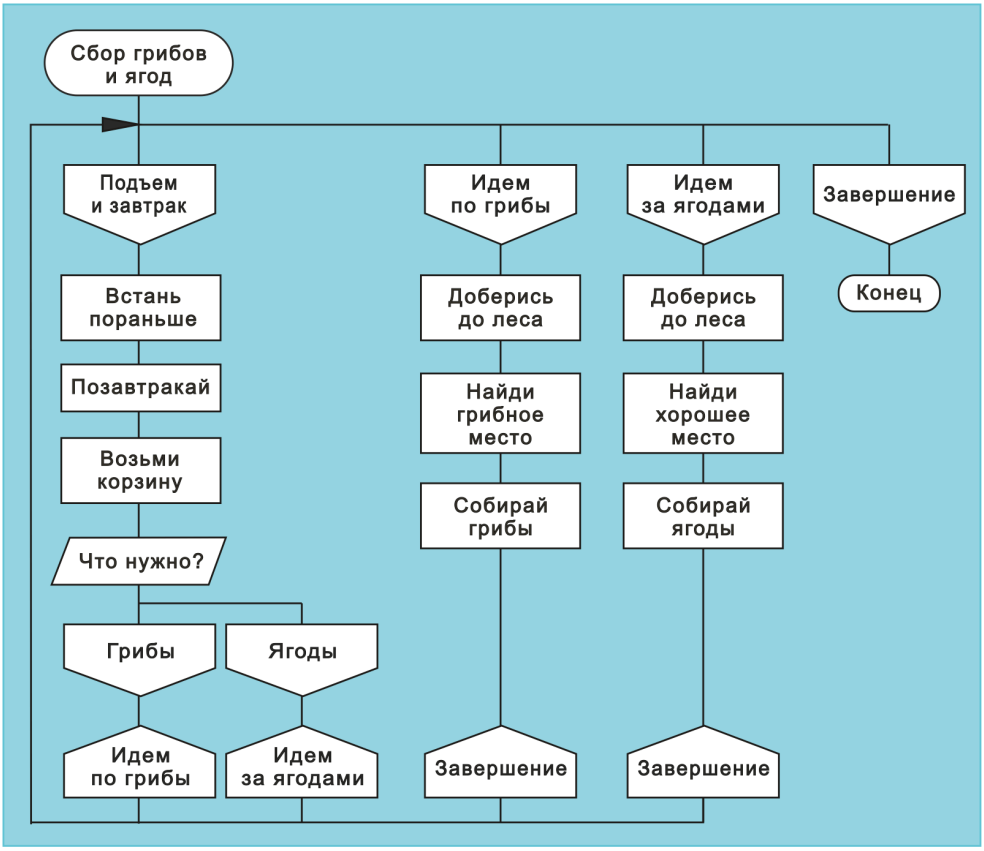


Рис. 92. Ветка «Подъем и завтрак» имеет два выхода

ПРАВИЛО ОДНОГО КОНЦА

Некоторые врачи по неопытности рисуют в алгоритме несколько концов. Это плохо.

Неприятность в том, что, глядя на схему, человеку трудно воспринимать алгоритм, имеющий несколько концов. Такой алгоритм похож на елку, увешанную «концами», как елочными игрушками. Подобная зрительная сцена распыляет внимание и мешает сосредоточиться на главном.

С эргономической точки зрения, желательно объединить все концы и превратить их в один. Это нетрудно сделать, как показано на рис. 92.

Чтобы объединить концы, в двух иконах Адрес написано слово «Завершение». Это слово означает прыжок в начало ветки «Завершение». Благодаря этому два «конца», уши которых торчат во второй и третьей ветках, объединяются и переносятся в последнюю ветку. Такой искусственный прием очень полезен – он позволяет соблюдать правило, запрещающее иметь в алгоритме несколько концов.

КАК СЛЕДУЕТ РАСПОЛАГАТЬ ВЕТКИ НА ЧЕРТЕЖЕ

Ветки упорядочены двояко: логически и в пространстве. *Логическая* последовательность исполнения веток определяется подсказками, записанными в иконах Адрес.

Однако логический порядок – это еще не все. Очень важно правильно расположить ветки в пространстве. Как это сделать?

Давайте мысленно перетасуем ветки, как колоду карт (рис. 91). И расположим их на чертеже в произвольном порядке. Легко сообразить, что подобное «перепутывание» веток никак не отразится на работе алгоритма. Ведь очередность работы веток зависит только от икон Адрес. И совсем не зависит от расположения веток на листе бумаги. Словом, сколько ветки ни тасуй, получим тот же самый алгоритм.

Здесь есть тонкость. Перестановка веток не отражается на правильности алгоритма. Однако алгоритм должен быть не только правильным, но и понятным, эргономичным. Хаотичное расположение веток затрудняет понимание. А это недопустимо.

Поэтому нужно обязательно упорядочить ветки в пространстве чертежа. Удобнее всего расположить их слева направо в той последовательности, в какой они включаются в работу. Для этого служит правило: «Чем правее, тем позже». Оно означает: чем правее находится ветка, тем позже она работает.

ЧТО МЫ УЗНАЛИ В ЭТОЙ ГЛАВЕ

Мы познакомились с силуэтом, наиболее важным, удобным и широко применяемым инструментом языка ДРАКОН.

В силуэте появились две новые иконы: Имя ветки и Адрес. Таким образом, золотая десятка ДРАКОНа, с которой мы впервые повстречались на рис. 35, превратилась в золотую дюжину (рис. 93).

Преимущество в том, что один силуэт позволяет заменить несколько примитивов. Предположим, что алгоритм состоит из семи примитивов, логически связанных между собой. Семь примитивов можно легко превратить в семь веток, а последние объединить в единый силуэт.

Как превратить примитив в ветку? Очень просто.

Сначала надо заменить икону Заголовок примитива на икону Имя ветки. А затем вместо иконы Конец примитива подставить икону Адрес. Вот и все. Разумеется, превращение группы примитивов в один силуэт возможно лишь тогда, когда для этого имеются логические предпосылки.

В состав золотой дюжины входят десять икон (1–10) и две макроиконы (11 и 12).

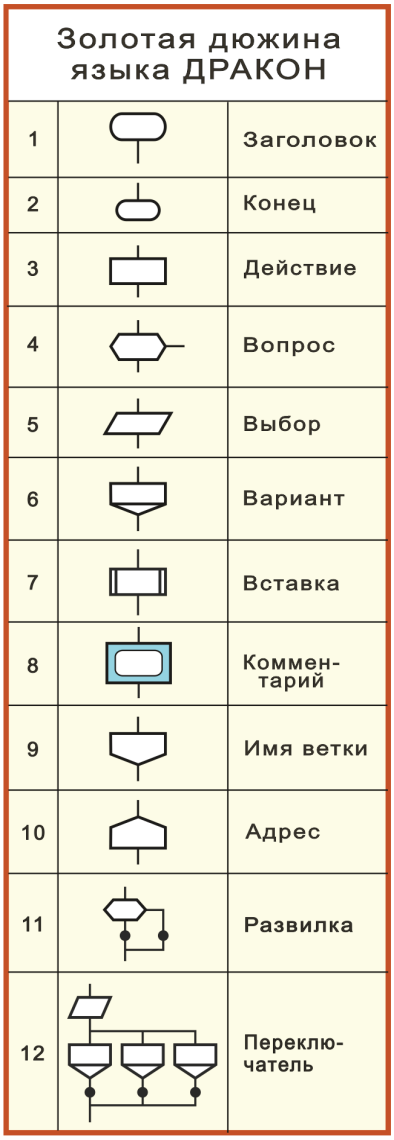


Рис. 93. Двенадцать фигур, часто встречающихся в медицинких алгоримах

ВЫВОДЫ

1. Существуют два типа дракон-алгоритмов: примитив и силуэт.
2. Примитив предназначен для описания простых медицинских алгоритмов, силуэт – для сложных и сверхсложных.
3. Силуэт состоит из веток.
4. Ветка содержит две новые иконы (Имя ветки и Адрес), которые отсутствуют в примитиве.
5. Икона Имя ветки обозначает начало ветки.
6. Икона Адрес обозначает конец любой ветки, кроме последней.
7. Ветки позволяют
   - разбить медицинский алгоритм на смысловые части,
   - дать частям удобные и точные названия.

8. Шапка – верхняя часть силуэта, которая включает Заголовок алгоритма и комплект икон Имя ветки.
9. Шапка позволяет читателю быстро сориентироваться в задаче и, расчленив ее на части, увидеть структуру алгоритма.
10. Ветки нужно упорядочить слева направо по принципу: «Более правая ветка работает позже, чем любая ветка, находящаяся левее нее» (кроме веточных циклов).

Глава 12

СЛОЖНЫЕ МЕДИЦИНСКИЕ АЛГОРИТМЫ. СИЛУЭТ. ПРАВИЛА И ПРИМЕРЫ

МЕДИЦИНСКИЕ ПРИМЕРЫ

В прошлой главе мы получили общее представление о силуэте. В этой главе речь пойдет о применении силуэта в медицине.

Алгоритмическая конструкция силуэт позволяет четко описать наиболее сложные особенности медицинской деятельности, включая тончайшие нюансы и мельчайшие подробности. Силуэт способен удовлетворить разнообразные требования и пожелания врачей, возникающие при разработке и использовании сложных и сверхсложных медицинских алгоритмов.

АЛГОРИТМ СИЛУЭТ «СНЯТИЕ ШЛЕМА С МОТОЦИКЛИСТА»

Для начала рассмотрим вопрос: как превратить примитив в силуэт?

Возьмем живой пример и превратим уже знакомый нам примитив на рис. 82 в силуэт, изображенный на рис. 94.

В чем недостаток примитива? Он не показывает структуру алгоритма. В самом деле, примитив на рис. 82 можно разбить на две части:

- Обеспечение стабильности головы.
- Снятие шлема.

Можно ли заметить эти части, глядя на рис. 82? Нет, нельзя. Нужно долго и мучительно соображать, чтобы догадаться, что граница между частями проходит между вторым и третьим синхронными участками (ниже иконы «Рукой охвати затылок»).

В силуэте на рис. 94 картина радикально меняется. В верхней части появились четкие указатели – две иконы Имя ветки. Словно яркие рекламные заголовки, они предъявляют глазу названия структурных частей алгоритма.

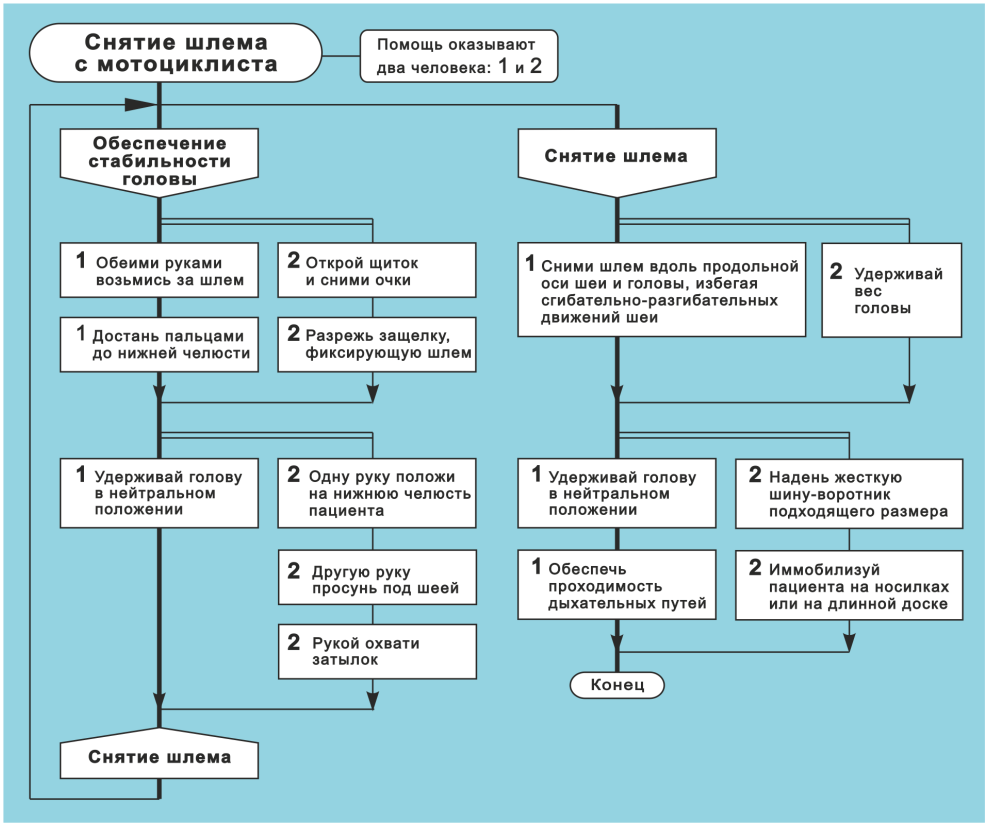


Рис. 94. Алгоритм силуэт, описывающий работу двух врачей [142]

КАК ЧИТАТЬ АЛГОРИТМ СИЛУЭТ

На рис. 94 надписи нужно читать сверху вниз вдоль левого шампура (жирная линия). Шампур кончается в иконе Адрес «Снятие шлема».

Куда идти дальше? Подсказка живет внутри самой иконы Адрес. По правилам ДРАКОНа такое же название должно быть наверху – в одном из рекламных заголовков.

Переведите взгляд наверх, найдите икону «Снятие шлема» и спуститесь вниз по правому шампуру до конца.

РАЗДЕЛЯЙ И ВЛАСТВУЙ. ВЕТКИ ОБЛЕГЧАЮТ ПОНИМАНИЕ

Наша цель – облегчить зрительное восприятие и анализ сложных медицинских алгоритмов. Спросим себя: можно ли сделать алгоритм обо-

зримым и удобным для быстрого восприятия, обдумывания и осмысления?

Можно. Чтобы любой врач смог легко прочитать и быстро понять алгоритм, надо заблаговременно разделить его на смысловые части. Сделать это нетрудно – надо использовать конструкцию силуэт. Преимущество в том, что силуэт заранее нарезан на удобные ломтики, предназначенные для быстрого схватывания и понимания.

Каждый такой ломтик (ветка) представляет собой довольно крупный кусок алгоритма. В составе ветки обычно несколько икон.

На рис. 94 две ветки, одна слева, другая справа. Первая озаглавлена «Обеспечение стабильности головы», она содержит 10 икон. Вторая носит имя «Снятие шлема» и имеет 8 икон.

Каждая ветка имеет свой шампур.

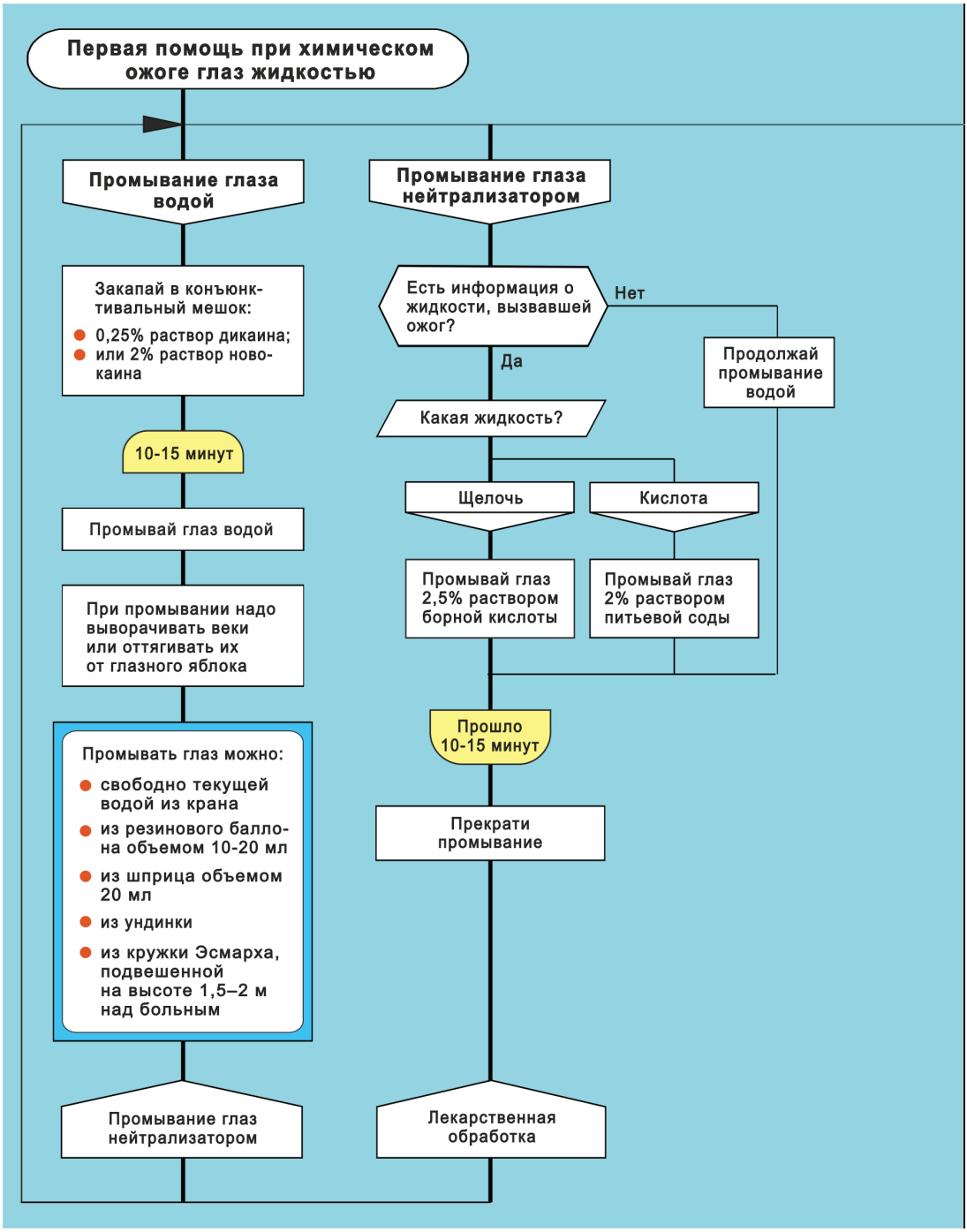
АЛГОРИТМ-СИЛУЭТ «ПЕРВАЯ ПОМОЩЬ ПРИ ХИМИЧЕСКОМ ОЖОГЕ ГЛАЗ ЖИДКОСТЬЮ»

Перейдем к следующему примеру (рис. 95, 96). Это алгоритмический рассказ, в котором можно выделить три части, три самостоятельных темы:

1. Промывание глаз водой.
2. Промывание глаз нейтрализующим раствором.
3. Лекарственная обработка.

Каждая тема нарисована в виде ветки.

Рис. 95. Шапка раскрывает структуру силуэта на рис. 96


Первая помощь при химическом ожоге глаз жидкостью
Промывание глаза водой
Закапай в конъюнктивальный мешок:
0,25% раствор дикаина;
или 2% раствор новокаина
10-15 минут
Промывай глаз водой
При промывании надо выворачивать веки или оттягивать их от глазного яблока
Промывать глаз можно:
свободно текущей водой из крана
из резинового баллона объемом 10-20 мл
из шприца объемом 20 мл
из ундинки
из кружки Эсмарха, подвешенной на высоте 1,5–2 м над больным
Промывание глаз нейтрализатором
Промывание глаза нейтрализатором
Есть информация о жидкости, вызвавшей ожог?
Нет
Да
Продолжай промывание водой
Какая жидкость?
Щелочь
Кислота
Промывай глаз 2,5% раствором борной кислоты
Промывай глаз 2% раствором питьевой соды
Прошло 10-15 минут
Прекрати промывание
Лекарственная обработка

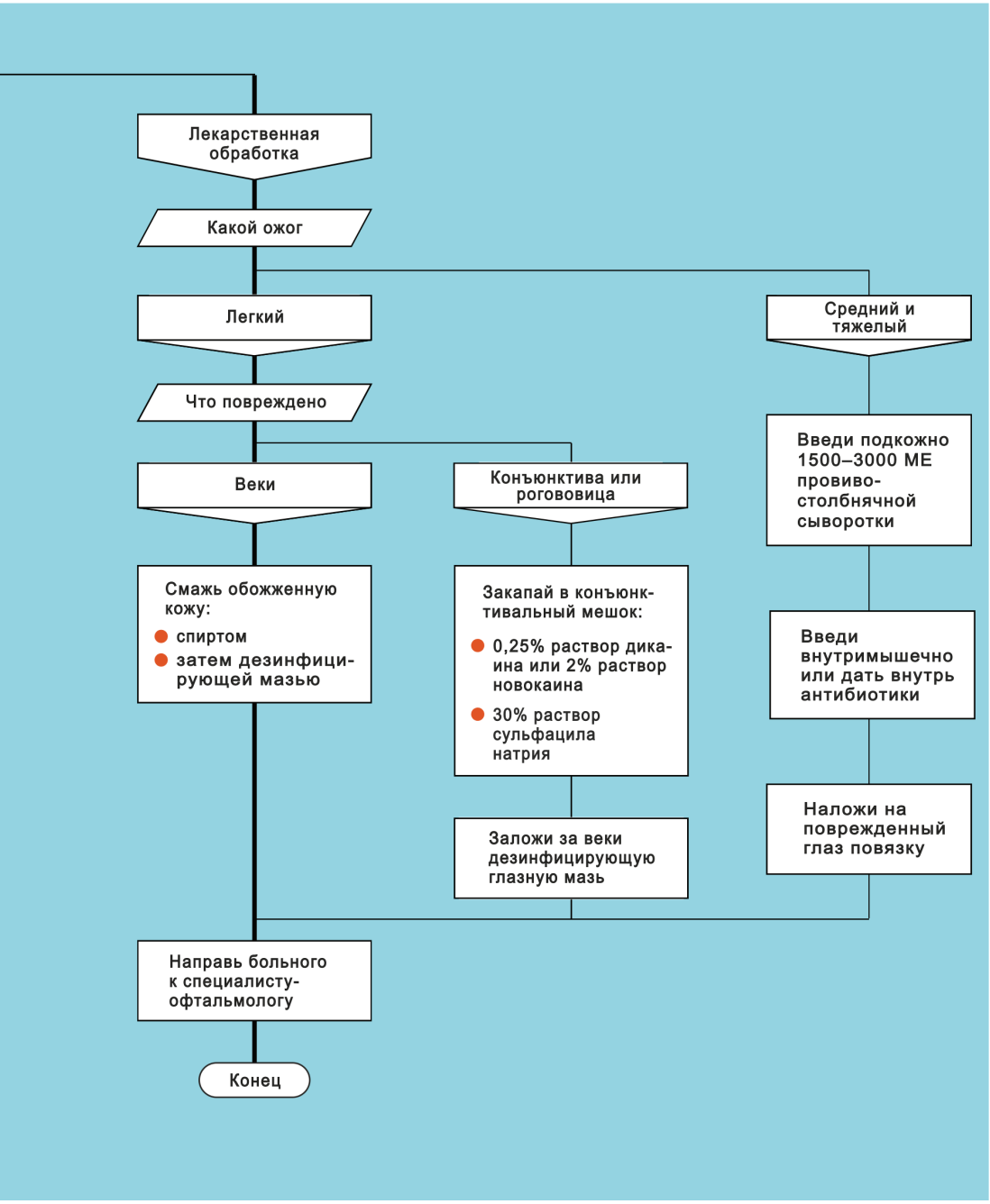


Рис. 96. Алгоритм «Первая помощь при химическом ожоге глаз жидкостью»

ЦАРСКИЕ ВОПРОСЫ

Вспомним наши царские вопросы:

1. Как называется задача?
2. Из скольких частей она состоит?
3. Как называется каждая часть?

Сложный алгоритм можно сравнить с лабиринтом. Чтобы не заблудиться, нужен план лабиринта. Шапка и есть такой план, путеводитель по лабиринту. Изучая шапку, получим ответы для алгоритма на рис. 96.

- Как называется задача? (*Читаем Заголовок алгоритма*).

  Первая помощь при химическом ожоге глаз жидкостью.

- Из скольких частей она состоит? (*Считаем иконы «Имя ветки»*).

  Из трех.

- Как называется каждая часть? (*Читаем текст в иконах «Имя ветки»*).

  1. Промывание глаз водой.
  2. Промывание глаз нейтрализующим раствором.
  3. Лекарственная обработка.

ШАПКА ПРИКОВЫВАЕТ К СЕБЕ ВНИМАНИЕ

Дополнительные удобства связаны с тем, что шапка занимает «парадное» место в верхней части чертежа. Названия смысловых частей помещены внутри особых рамок уникальной формы, которые легко отыскать взглядом.

Благодаря этому шапка моментально приковывает к себе внимание читателя без всяких усилий с его стороны. Поэтому человеку не приходится рыскать глазами по темным закоулкам алгоритма, пытаясь выудить нужную информацию.

ЧИТАЕМ ПЕРВУЮ ВЕТКУ

Вопрос. Где начало и конец у первой ветки на рис. 96?

Ответ. Начало – икона «Промывание глаз водой». Конец – «Промывание глаз нейтрализующим раствором». Между началом и концом размещается тело ветки, которое содержит пять икон:

- «Закапай в конъюнктивальный мешок...».
- «10–15 минут».
- «Промывай глаз водой».

- «При промывании надо выворачивать веки...».
- «Промывать глаз можно...».

Вопрос. Зачем нужна икона Адрес «Промывание глаз нейтрализующим раствором»?

Ответ. Икона Адрес говорит: «Прыгни в икону “Имя ветки” с таким же названием». Смысл в том, что после выполнения ветки «Промывание глаз водой» будет выполняться ветка «Промывание глаз нейтрализующим раствором».

ЧИТАЕМ ВТОРУЮ ВЕТКУ

Вторая ветка разветвленная, в ней три маршрута. Прочитаем главный из них, который идет по шампуру (рис. 96).

Начало ветки – икона «Промывание глаз нейтрализующим раствором». Конец – «Лекарственная обработка». Между началом и концом находятся иконы:

- «Есть информация о жидкости...?» Да.
- «Какой маршрут?».
- «Щелочь».
- «Промывай глаз 2,5% раствором...».
- «Прошло 10–15 минут».
- «Прекрати промывание».

В самом низу виднеется икона Адрес «Лекарственная обработка». Она говорит: «Прыгни в икону “Имя ветки” с таким же названием». Это значит, что после выполнения ветки «Промывание глаз нейтрализующим раствором» будет выполняться следующая ветка – «Лекарственная обработка».

Оставшиеся два маршрута выполняются аналогично.

ЧИТАЕМ ТРЕТЬЮ ВЕТКУ

Третья ветка также имеет три маршрута. Пройдемся по главному маршруту (рис. 96).

В начале ветки лежит икона «Лекарственная обработка». В самом низу икона «Конец» – делу венец. Между началом и концом нарисованы шесть элементов:

- «Какой ожог».
- «Легкий».
- «Что повреждено».
- «Веки».
- «Смажь обожженную кожу».
- «Направь больного к специалисту».

Боковые маршруты третьей ветки мы пропускаем, поскольку они очень просты.

ДРУГОЙ СПОСОБ ОПИСАНИЯ СИЛУЭТА

Силуэт можно описывать двумя способами. Выше мы продемонстрировали метод обычного чтения.

Второй способ – метод бегунка. Вспомним, что воображаемый бегунок при работе алгоритма пробегает алгоритмическую дорожку от иконы Заголовок до иконы Конец.

Как работает алгоритм на рис. 96? Выехав из иконы Заголовок, бегунок мчится вниз по крайней левой ветке. Он движется через станции:

- «Промывание глаз водой».
- «10–15 минут».
- и т. д.

Затем он попадает в икону Адрес – последнюю станцию первой ветки. Куда ехать дальше? Ответ прячется в самой иконе. Эта икона потому и зовется «Адрес», что сообщает адрес (название) следующей станции. Бегунок узнает, что он должен попасть на станцию «Промывание глаз нейтрализующим раствором».

Данная станция находится в начале второй ветки. Но как туда попасть? По какой линии?

Ответ известен. Выехав из иконы Адрес, бегунок резко берет влево и по внешней обрамляющей линии огибает всю схему. Потом едет направо по стрелке и попадает в верхнюю икону «Промывание глаз нейтрализующим раствором».

Дальше события развиваются, как обычно. Бегунок скользит вниз по второй ветке. Добравшись до иконы Адрес, узнает адрес нужной станции («Лекарственная обработка»). Затем огибает схему по часовой стрелке, попадает в начало третьей ветки и спускается по ней до самого конца. На этом наш рассказ завершается.

КОНТРОЛЬНОЕ ВРЕМЯ ПРОЦЕДУРЫ

Многие медицинские действия и решения привязаны ко времени. Типичным примером является так называемое контрольное время. На рис. 96–98 оно показано в виде двух икон желтого цвета:

- Начало контрольного срока.
- Конец контрольного срока.

В руководстве длительность промывания глаз при химическом ожоге строго определена:

«Промывание водой или нейтрализующими растворами продолжают в течение 10–15 минут, чтобы удалить из конъюнктивального мешка ту часть повреждающего агента, которая еще не вступила в соединение с тканями» [144].

В алгоритме указанную длительность рассматривают как *контрольное время*, которое должно быть задано в виде двух точно определенных моментов времени: начального и конечного. При этом следует различать два случая:

- моменты находятся в разных ветках (рис. 97, 98),
- оба момента расположены в одной ветке (рис. 102, 105).

В первом случае используют иконы «Начало контрольного срока» и «Конец контрольного срока».

В нашем примере так и сделано. Икона «Начало контрольного срока» поставлена в первой ветке перед иконой «Промывай глаз водой» (рис. 96, 97). А «Конец...» помещен во второй ветке выше иконы «Прекрати промывание» (рис. 96, 98).

Учтите зрительную подсказку: икона Начало срока похожа на верхнюю полусферу, икона Конец срока – на нижнюю полусферу.

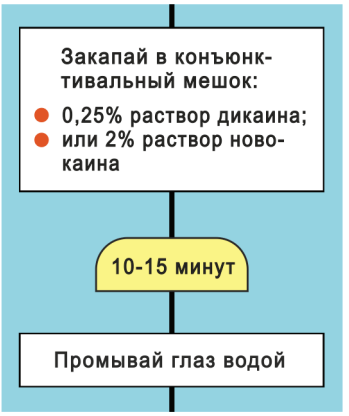


Рис. 97. Икона «Начало контрольного срока»

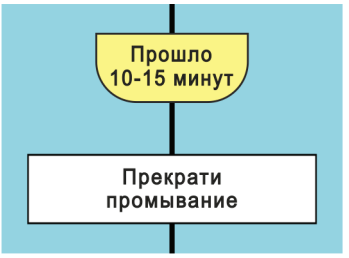


Рис. 98. Икона «Конец контрольного срока»

МАРШРУТЫ ВЕТКИ

Что такое шампур ветки?

Это вертикаль, соединяющая икону «Имя ветки» с иконой Адрес, а если ветка имеет несколько икон Адрес – с левой из них.

Каждая ветка силуэта имеет свой шампур. На рис. 96 три ветки. Следовательно, данный силуэт имеет три шампура (три жирных вертикали).

Для ветки сохраняют силу оба царских правила:

- главный маршрут ветки должен идти по шампуру ветки;

- все маршруты ветки следует упорядочить по принципу: «Чем правее, тем хуже».

Последнее правило поясним на примере ветки «Лекарственная обработка» на рис. 96. Она имеет три вертикали.

Левая вертикаль (главный маршрут ветки) описывает наименьший ущерб для пациента, так как ожог легкий, причем обожжены только веки, а наиболее нежные ткани (конъюнктива и роговица) не пострадали.

Средняя вертикаль занимает промежуточное положение. Речь идет о легком ожоге, но, к сожалению, обожженными оказались конъюнктива и роговица.

Правая вертикаль означает наихудшую ситуацию, описывающую средний или даже тяжелый химический ожог.

Вопрос. Сколько маршрутов в силуэте на рис. 96?

Ответ. Во второй ветке 3 маршрута, в третьей тоже 3. Следовательно, общее число маршрутов в силуэте равно $3 \times 3 = 9$.

Правило маршрутов ветки	Чем правее (чем дальше от шампура данной ветки) расположена очередная вертикаль, тем более неприятную для пациента ситуацию она описывает.
Как выполняются действия внутри ветки?	Действия внутри ветки выполняются по принципу: «Чем ниже, тем позже» (при отсутствии циклов).

СВОЙСТВА ВЕТКИ

Каждая ветка силуэта (рис. 96) имеет следующие свойства.

- Вверху врач видит начало ветки, внизу – конец ветки.
- *Вход* в ветку возможен только через ее начало.
- *Выходом* из ветки служит икона Адрес, в которой записывается имя следующей по порядку исполнения ветки.
- Выход из последней ветки осуществляется через икону Конец.

ЧТО БУДЕТ, ЕСЛИ УБРАТЬ ОБРАМЛЕНИЕ

Понятие *обрамление* в силуэте включает четыре элемента:

- верхнюю горизонталь (верхнюю шину),
- нижнюю горизонталь (нижнюю шину),
- левую вертикальную линию, соединяющую их,
- маленькую стрелку в левом верхнем углу.

Вспомним еще раз чеховского героя, уберем обрамление из алгоритма на рис. 96 и представим полученный чертеж на рис. 99.

Мы видим любопытную картину. Целостный алгоритм распался на три изолированные части, похожие на три острова в океане. Связи между ними полностью исчезли! Острова разделены четкими синими проливами, причем мосты между островами отсутствуют.

Мы знаем, однако, что все эти украшения носят чисто художественный, декоративный, эстетический характер. Отсутствие обрамления никак не влияет на работу алгоритма. Причина нам также хорошо известна. Взаимодействие между ветками силуэта объясняется не художественными изысками обрамления, а невидимыми логическими связями между «островами». В качестве таких связей выступают указания, записанные в иконах Адрес. Именно эти указания являются невидимым клеем, который скрепляет между собой разрозненные ветки силуэта. И превращает их в единый монолитный, слаженно действующий ансамбль.

Рис. 99 помогает уяснить две истины:

- Связи между ветками возможны только через иконы Адрес.
- Любые иные связи между ветками запрещены.

Смысл запрета в том, что соединительные линии между ветками объявляются незаконными. Если кто-то проведет явную линию между ветками (отличную от линий обрамления), это рассматривается как грубая ошибка. Такая ошибка кажется незаметной на рис. 96. Но на рис. 99 она будет сразу выявлена – как попытка построить пиратский мост между островами.

Завершая этот параграф, следует еще раз подчеркнуть, что линии обрамления нужны по эргономическим соображениям. Они, словно рама у картины, соединяют, сочленяют разрозненные части зрительной сцены. Они превращают их в приятный для глаза целостный зрительно-смысловой образ. Изъятие скрепляющих линий недопустимо, оно приводит к негативным последствиям – схема зрительно рассыпается на части, что сбивает с толку читателя (рис. 99). Кроме того, скрепы помогают проследить любой маршрут пальцем, не отрывая его от бумаги или экрана.

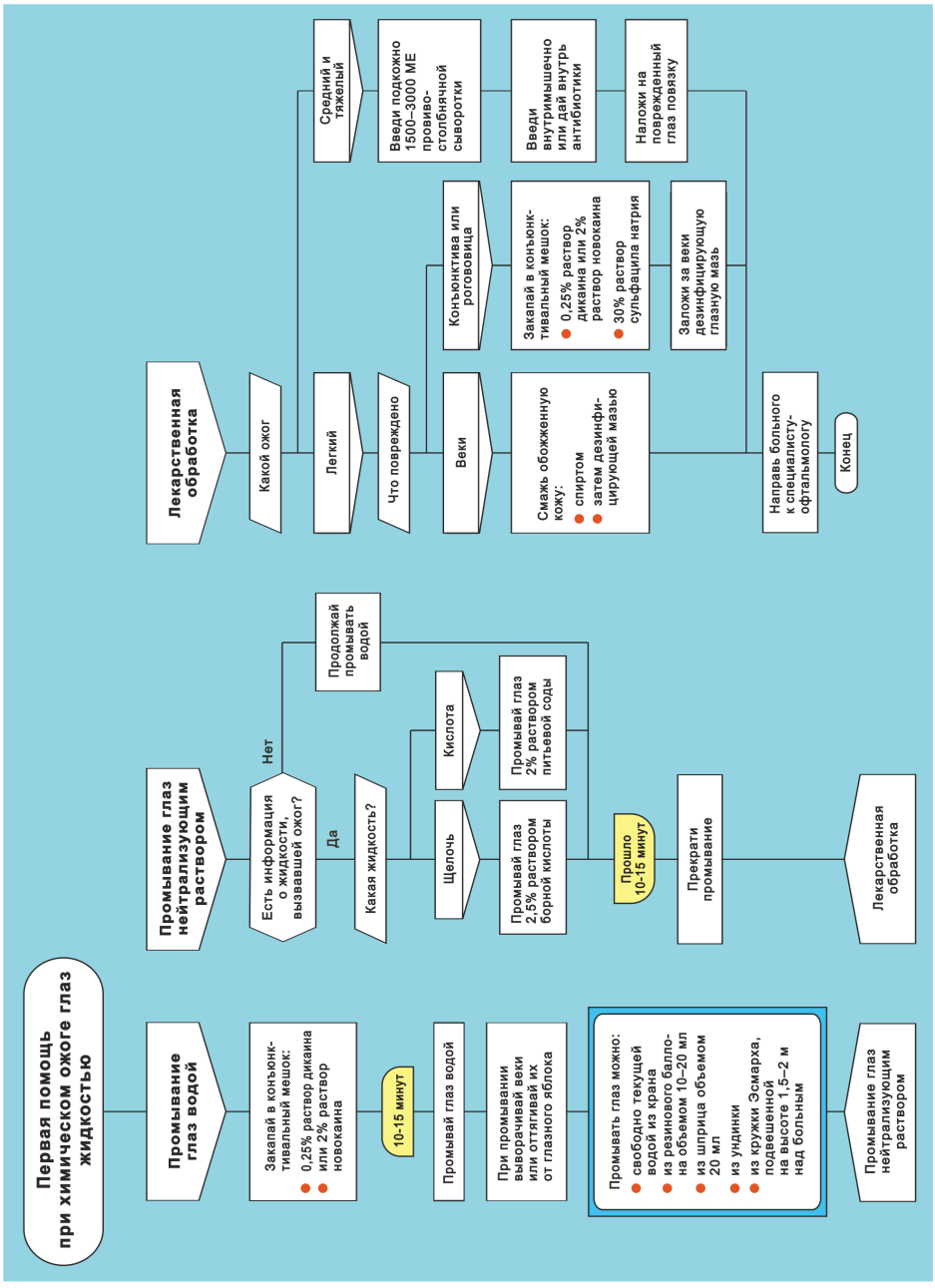


Рис. 99. Алгоритм с удаленными обрамляющими линиями. Сравните с рис. 96

ВЫВОДЫ

1. Алгоритм силуэт позволяет описать наиболее сложные особенности медицинской деятельности.
2. Примитив не показывает структуру, а силуэт наглядно демонстрирует структурные части алгоритма.
3. Силуэт позволяет облегчить зрительное восприятие и анализ сложных медицинских алгоритмов.
4. Силуэт заранее расчленен на удобные структурные элементы (ветки), предназначенные для быстрого схватывания и понимания.
5. Все маршруты силуэта начинаются в иконе Заголовок и кончаются в иконе Конец
6. Если границы контрольного времени находятся в разных ветках, используются иконы «Начало контрольного срока» и «Конец контрольного срока».
7. Шампур ветки – вертикаль, соединяющая икону Имя ветки с иконой Адрес, а если ветка имеет несколько Адресов, – с левой из них
8. Главный маршрут ветки должен идти по шампуру ветки.
9. Чем правее (чем дальше от шампура данной ветки) расположена очередная вертикаль, тем более неприятную для пациента ситуацию она описывает.
10. Действия внутри ветки выполняются по принципу: «Чем ниже, тем позже» (при отсутствии циклов).
11. Связи между ветками возможны только через иконы Адрес. Все остальные соединения между ветками запрещены.
12. Для наглядности между ветками желательно оставлять хорошо различимые воздушные промежутки.

Глава 13

КАРТОГРАФИЧЕСКИЙ ПРИНЦИП МЕДИЦИНСКОГО АЛГОРИТМА (ПРИНЦИП КРАСОТЫ)

ЧТО ДУМАЮТ УЧЕНЫЕ О КРАСОТЕ. КРАСОТА КАК ЭРГОНОМИЧНОСТЬ

Понятие «красота» в науке можно трактовать по-разному. Существует мнение, что красота – это совершенство научных результатов, вызывающее восхищение у научного сообщества. Такое восхищение чем-то сродни преклонению перед волшебством и тайнами поэзии. «Нельзя быть математиком, не будучи в то же время и поэтом в душе», – считает немецкий математик Карл Вейерштрасс [145].

Несколько иначе понимает красоту известный физик Поль Дирак:

> «Общие законы природы, когда они выражены в математической форме, обладают математической красотой... Это дает физику-теоретику могучий метод, руководящий его действиями. Если он видит, что в его теории есть уродливые части, то он считает, что именно эти части неправильны и он должен сконцентрировать на них свое внимание [чтобы их улучшить]... Этот прием изыскания математического изящества является наиболее существенным для теоретиков» [146].

Иногда говорят, что красота отражает эмоциональную, психологическую восторженность человека, вызванную наличием гармонии, изящества, неординарности в продуктах научного творчества [147].

Но есть и другая точка зрения. Физик Леон Николя Бриллюэн полагает, что красота выражает «приятное ощущение понимания и постижения». Для наших целей позиция Бриллюэна представляет большой интерес, так как она позволяет трактовать красоту как эргономичность.

КРАСОТА АЛГОРИТМОВ

> Алгоритм, представленный в письменном виде, предназначен для зрительного восприятия человеком. Следовательно, алгоритм представляет собой зрительную сцену. Или, если угодно, зрительный образ, зрительную картину.

Красота алгоритма – это красота его зрительного образа, красота дракон-схемы.

Алгоритм можно назвать *красивым* (эргономичным), если процесс зрительного восприятия, понимания и постижения алгоритма протекает с максимальной скоростью, наименьшими усилиями и максимальным эстетическим наслаждением.

Чем красивее алгоритм, тем быстрее и легче можно его понять. Отсюда вытекает, что красота и элегантность алгоритмов открывают путь к экономии умственных усилий. Но не только.

> Чем красивее зрительные образы отдельных частей алгоритма, чем изящнее они соединены в общую алгоритмическую картину, тем приятнее на них смотреть. Чем точнее и элегантнее зрительный «пейзаж» обнажает глубинный смысл алгоритма, тем плодотворнее мышление.

Чем больше красоты, тем глубже понимание алгоритмов. Тем скорее течет наша алгоритмическая мысль. Тем легче мы постигаем суть дела. Тем быстрее и качественнее протекает процесс создания медицинских алгоритмов.

И наоборот, если зрительный образ алгоритма кажется некрасивым, неприятным, отталкивающим и запутанным, процесс понимания и обдумывания неизбежно замедляется, что снижает производительность умственного труда.

ДРАКОН – графический язык, язык зрительных образов. С учетом сказанного, можно уточнить: ДРАКОН – язык красивых (эргономичных) зрительных образов. Но красота ДРАКОНа – не самоцель. Она позволяет ощутимо повысить производительность труда при разработке, изучении и понимании медицинских алгоритмов.

Мы исходим из того, что зрительные образы алгоритмов следует сознательно проектировать. Для этой цели можно (в разумных пределах) использовать *средства художественного конструирования*.

Уместно напомнить слова видного психолога Бориса Ломова (создателя Института психологии Академии наук СССР):

> «Средства художественного конструирования в конечном счете направлены на то, чтобы вызвать тот или иной эффект у работающего человека...
>
> Применяя средства художественного конструирования, мы создаем положительные эмоции, облегчаем операцию приема информации человеком, улучшаем концентрацию и переключение внимания, повышаем скорость и точность действий.
>
> Короче говоря, мы пользуемся этими средствами для управления поведением человека в широком смысле слова, для управления его психическим состоянием» и умственной работоспособностью» [235].

Это небольшое вступление, посвященное красоте и эргономичности научных знаковых систем, призвано пролить дополнительный свет на основной принцип силуэта, который представляет собой *принцип географической карты*, или картографический принцип. При дальнейшем изложении мы избегаем рассуждений о красоте, однако ее отблески, ее призывный свет несомненно подразумеваются.

КАРТОГРАФИЧЕСКИЙ ПРИНЦИП СИЛУЭТА

Картографический принцип говорит о том, что движение взора по географической карте имеет большой смысл. Глаза учитывают и отслеживают два направления: север – юг и запад – восток.

Точно так же перемещение взгляда по алгоритму должно иметь строго определенный смысл. Движение глаз вверх-вниз (по вертикали) и влево-вправо (по горизонтали) не должно быть безразличным и бессмысленным. Наоборот, оно должно нести важную информацию и, проникая через хрусталик глаза, обогащать наш мозг ценными сведениями.

В главе 7 мы описали картографический принцип примитива. Он означает, что слева находятся более благоприятные для пациента (хорошие) маршруты. Справа – менее благоприятные (плохие). По вертикали смысл такой: вверху начало времени, внизу – конец.

Следует учесть, что примитив – аналог ветки. Следовательно, картографический принцип примитива полностью сохраняет силу для ветки.

Оговоримся. Принцип для ветки имеет локальный характер; он действует внутри одной ветки. И не распространяется за ее пределы.

Сделаем следующий шаг и определим порядок взаимодействия веток. И тем самым распространим картографический принцип на весь силуэт.

Для этого нужно упорядочить ветки в пространстве. Лучше всего расположить их слева направо в той последовательности, в какой они включаются в работу. Для этого служит правило: «Чем правее, тем позже». Оно означает: чем правее находится ветка, тем позже она работает.

Силуэт, нарисованный согласно правилу «Чем правее, тем позже», считается красивым, эргономичным (рис. 96). Схемы, где это правило нарушается, объявляются плохими. Их использование запрещено.

Правило построения веток в силуэте

Чтобы силуэт был красивым и удобным для работы, ветки нужно расположить слева направо согласно принципу: «Более правая ветка работает позже, чем любая ветка, находящаяся левее нее» (кроме веточных циклов).

В разрешенных (эргономичных) алгоритмах имеет место следующий порядок действий (рис. 96):

- первой работает крайняя левая ветка, последней – крайняя правая;
- остальные ветки передают эстафету друг другу слева направо (при этом может случиться так, что некоторые ветки будут пропущены).

Отсюда следует, что время в силуэте перемещается не по одной, а по двум осям – и по вертикали, и по горизонтали.

- Двигаясь по ветке сверху вниз, мы перемещается во времени от начального момента до конечного (на данном отрезке). Время движется по ветке вертикально вниз.
- Двигаясь по веткам слева направо, мы тоже перемещаемся во времени. По разным веткам время движется горизонтально слева направо (при отсутствии веточных циклов).

При этом никакой путаницы не возникает, все учтено и хорошо продумано в соответствии с принципом красоты. По одной ветке время бежит всегда вниз, по разным веткам – всегда направо (кроме циклов).

МОЖНО ЛИ НАВЕСТИ ПОРЯДОК В МЕДИЦИНСКИХ АЛГОРИТМАХ

В некотором царстве, в тридесятом государстве были два города. Один строился без плана, вкривь и вкось: тесные улочки и переулки сплелись в

змеиный клубок (рис. 100). Зато другой был образцовым: красивые площади, широкие проспекты, всюду простор и порядок (рис. 101). В каком городе легче найти дорогу?

Конечно, во втором.

Рис. 100. Хаотическая планировка и запутанная застройка с кривыми улочками характерна для многих старинных городов, которые стихийно разрастались на месте древних поселков

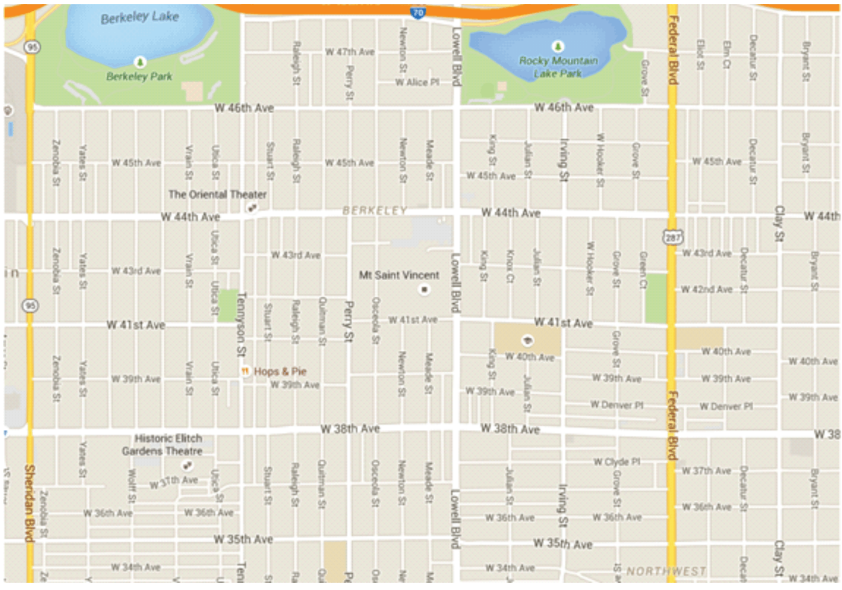


Рис. 101. Современное строительство города на равнине позволяет заранее оставить четкий план и строго его соблюдать. Все чинно-благородно, никаких нарушений, никакой партизанщины

А теперь представьте, что нам нужно изучить незнакомый медицинский алгоритм. Если в нем, как в первом городе, нет порядка, он превращается в запутанный лабиринт, к котором ничего нельзя понять. Если же он, как второй город, построен по хорошему плану, ситуация в корне меняется. Про такой город говорят: Красота – все ясно, как на ладони!

Чтобы убедиться в этом, давайте посмотрим еще разок на рис. 96 и проведем взглядом по всем вертикалям слева направо. Мы обнаружим не хаос, а строгий порядок. Потому что маршруты нарисованы не случайно, не как левая нога захочет, а по строгим правилам, правилам красоты.

КРАСИВОЕ И УРОДЛИВОЕ

Вспомним слова Поля Дирака: если в «теории есть уродливые части, то... именно эти части неправильны». Данную мысль можно применить и к медицинским алгоритмам. В восьмой главе мы видели, что операция рокировки легко превращает алгоритмического «урода» в «красавца». Язык ДРАКОН способен существенно улучшить эстетические (эргономические) характеристики медицинских алгоритмов.

ЧТО ЛУЧШЕ: БЛОК-СХЕМА ИЛИ ДРАКОН-СХЕМА?

Существует и другая точка зрения. Специалисты из Самарского медицинского университета рекомендуют рисовать медицинские алгоритмы в виде блок-схем согласно Государственному стандарту ГОСТ 19.701-90. В защиту своей позиции они приводят следующие аргументы:

> «Активное введение в учебный процесс... стандартных блок-схем и описываемых ими алгоритмов действий оправдано по трем причинам.
>
> - Во-первых, студенты, как правило, знакомятся с ними еще в школе, на занятиях по информатике.
> - Во-вторых, их активно вводят в практику врачи, защищающиеся по относительно новой для медицины специальности 05.13.01 «Системный анализ, управление и обработка информации».
> - В-третьих, они в методическом плане позволяют визуализировать ход принятия решений, его этапы и составляющие, причем делается это стандартными и уже знакомыми студентам символами, что облегчает восприятие как блок-схем, так и учебного материала» [148].

Предложение самарских коллег вызывает возражения, ибо указанный стандарт является калькой международного стандарта ISO 5807-85 тридцатилетней давности, который отстал от жизни, построен без учета современных когнитивно-эргономических идей и концептуально устарел.

Есть ли замена блок-схемам? Да. В 1996 г. Государственный комитет Российской Федерации по высшему образованию включил изучение языка ДРАКОН в программу курса «Информатика» для направлений:

510000 – Естественные науки и математика
540000 – Образование
550000 – Технические науки
560000 – Сельскохозяйственные науки [149].

В официальном документе Госкомвуза «Примерная программа дисциплины “Информатика”» имеются разделы, посвященные языку ДРАКОН[1] [150]. В программе, в частности, говорится:

> «Для решения столь масштабных задач нужен универсальный язык представления процедурных знаний в любой предметной области. Это должен быть язык нового типа: общедоступный, человечный, предельно легкий в изучении и удобный в работе..., удовлетворяющий самым строгим эргономическим и дидактическим требованиям... В наибольшей степени этим требованиям соответствует процедурный язык... ДРАКОН, являющийся обобщением опыта, накопленного при создании космического корабля Буран [151]».

ДРАКОН-СХЕМА – ЭТО КРАСИВАЯ, ПРАВИЛЬНО ПОСТРОЕННАЯ БЛОК-СХЕМА

Дракон-схемы построены на основе блок-схем с целью их совершенствования и развития.

Недостаток блок-схем заключается в том, что они не приучают к аккуратности при разработке алгоритма. Ромб можно поставить в любом месте блок-схемы, а от него повести выходы на какие угодно участки. «Так можно быстро превратить алгоритм в запутанный лабиринт, разобраться в котором через некоторое время не сможет даже сам его автор» [152].

Блок-схемы не позволяют изображать сложные алгоритмы с необходимой полнотой и наглядностью. Чтобы устранить дефект, нужно *упорядочить* блок-схемы. Упорядоченные блок-схемы называются «дракон-схе-

[1] «Примерная программа дисциплины “Информатика”» одобрена Президиумом совета по информатике Госкомвуза. Председатель Президиума академик РАН Юрий Журавлев – руководитель Секции прикладной математики и информатики Отделения математических наук РАН, заместитель Академика-секретаря Отделения математических наук РАН.

мы» и подчиняются строгим формальным правилам и правилам эргономичных алгоритмов [71].

В отличие от блок-схем, дракон-схемы пригодны для формализованной записи и автоматического получения исполняемого кода. Запрещено пересечение линий, которое путает читателей и затрудняет понимание алгоритма, устранены другие некрасивые места. Дракон-схемы позволяют ликвидировать или существенно ослабить недостатки блок-схем.

Дракон-схемы специально сконструированы таким образом, чтобы превратить сложный алгоритм в удобную схему, обеспечивающую быстрое и легкое понимание [153, 154]. По мнению специалистов, благодаря использованию дракон-схем алгоритмы становятся более понятными, доходчивыми, ясными, прозрачными. Эргономичные методы, применяемые в дракон-схемах, существенно улучшают восприятие алгоритмов. Язык ДРАКОН обеспечивает разработку сложных алгоритмов с сохранением наглядности даже для многостраничных схем [155, 156].

РЕКОМЕНДАЦИИ АВТОРАМ МЕДИЦИНСКИХ УЧЕБНИКОВ

В некоторых медицинских учебниках в качестве иллюстраций используются блок-схемы алгоритмов. Это ошибка, которая создает неоправданные трудности для студентов медицинских университетов. Ошибка, вызванная отсутствием достоверной информации. Ошибка, которую необходимо исправить.

Хотя стандарты на блок-схемы считаются действующими, фактически они давно устарели. С появлением дракон-схем блок-схемы полностью потеряли свое значение, так как они во всех отношениях уступают дракон-схемам [157].

Язык ДРАКОН был разработан, в частности, потому, что традиционные блок-схемы алгоритмов, с эргономической точки зрения, не выдерживают критики. Они напоминают непроходимые джунгли, в которых легко запутаться и сбиться с дороги.

ДРАКОН выгодно отличается тем, что его графический узор подчиняется жестким и тщательно продуманным правилам, которые дисциплинируют мышление, облегчают умственный труд [158].

Важным дефектом блок-схем является недостаток выразительных средств. Алгоритмическая структура силуэт, являющаяся основным и наиболее мощным инструментом языка ДРАКОН, принципиально не может быть выражена на языке блок-схем.

Изложенные в литературе теоретические обоснования, сравнительный анализ дракон-схем и блок-схем, а также многочисленные примеры убедительно доказывают, что блок-схемы не имеют будущего [71–73].

В связи с этим можно рекомендовать авторам медицинских учебников отказаться от блок-схем и использовать язык ДРАКОН как наиболее эффективное средство для записи медицинских алгоритмов, помогающее облегчить учебный процесс и повысить качество обучения.

ВЫВОДЫ

1. Медицинский алгоритм называется красивым (эргономичным), если процесс зрительного восприятия и осмысления алгоритма протекает с максимальной скоростью, наименьшими усилиями и доставляет удовольствие читателю.
2. Красивый алгоритм должен быть упорядоченным, пригодным для быстрого восприятия и запоминания.
3. Примитив – аналог ветки.
4. Картографический принцип примитива действителен для ветки.
5. Картографический принцип ветки гласит:
   - в левой части ветки находятся более благоприятные для пациента вертикальные маршруты, в правой – менее благоприятные.
   - ориентация ветки по вертикали такова: вверху начало времени, внизу – конец.
6. Картографический принцип ветки имеет локальный характер и не распространяется за ее пределы.
7. Картографический принцип силуэта гласит:
   - в каждой ветке время направлено вертикально вниз;
   - внутри ветки вертикальные маршруты упорядочены слева направо от наиболее благоприятных для пациента до наименее благоприятных;
   - в силуэте время движется по горизонтали слева направо;
   - первой работает крайняя левая ветка, последней – крайняя правая.
8. В медицинских алгоритмах на языке ДРАКОН все маршруты тщательно упорядочены и организованы в красивую зрительную сцену, предназначенную для облегчения изучения и запоминания алгоритмов.
9. Блок-схемы алгоритмов безнадежно устарели. Пользоваться ими недопустимо. Вместо них следует использовать дракон-схемы.

Глава 14

АЛГОРИТМ «РЕАНИМАЦИЯ БЕРЕМЕННОЙ ЖЕНЩИНЫ»

НОВЫЕ ВОЗМОЖНОСТИ

Мы успешно преодолели подготовительные ступени при изучении ДРАКОНа и забрались почти на самую вершину. Осталось сделать последний рывок и овладеть увлекательной темой, которая называется «Многоадресный силуэт». Имеется в виду силуэт, содержащий *многоадресные* ветки.

До сих пор мы знакомились с простыми случаями, когда все ветки силуэта имеют только одну икону Адрес (рис. 87, 94, 96). Силуэт с многоадресными ветками обладает новыми многообещающими возможностями:

- можно изменять последовательность выполнения веток,
- можно выборочно запускать ветки, пропуская одну из них и даже несколько,
- можно повторять выполнение веток, т. е. создавать веточные циклы.

Более широкие возможности позволяют существенно улучшить качество медицинских алгоритмов, сделать их более точными и наглядными.

Далее мы рассмотрим поучительный пример, разъясняющий идею многоадресного силуэта. План изложения такой.

- Сначала взглянем на общую картину силуэта.
- Потом выявим структуру с помощью шапки.
- После этого рассмотрим ветки поочередно слева направо, задерживаясь на всех интересных местах.

МНОГОАДРЕСНЫЙ СИЛУЭТ «РЕАНИМАЦИЯ БЕРЕМЕННОЙ ЖЕНЩИНЫ»

Существует ряд причин, которые могут вызвать внезапную (клиническую, т. е. обратимую) смерть беременной женщины. На рис. 102 показан алгоритм реанимации для этого случая. Силуэт делится на шесть смысловых

частей – веток. Каждую ветку читают сверху вниз. Разные ветки изучают слева направо, учитывая указания в иконах Адрес.

Начинать чтение нужно с первой, крайней левой ветки, постепенно перемещаясь направо до конца.

Силуэт называется многоадресным, если он содержит многоадресные ветки. На рис. 102 видно, что вторая и третья ветки являются многоадресными, они имеют по три иконы Адрес. По этой причине силуэт на рис. 102 является многоадресным.

ВАЖНЫЕ ВЕЩИ НУЖНО ВЫДЕЛЯТЬ. КАК ЭТО СДЕЛАТЬ?

На рис. 102 взору зрителя одновременно представлено большое количество информации, буквально глаза разбегаются. Если человек впервые видит столь сложную зрительную сцену, он может испытывать дискомфорт и ощущать неудовольствие. Действительно, зрительная картина содержит 42 иконы и множество соединительных линий. Вся эта лавина информации разом обрушивается на неподготовленного зрителя, и человек поневоле теряется.

Чтобы этого не случилось, в языке ДРАКОН предусмотрены специальные средства, позволяющие управлять движением глаз наблюдателя и концентрировать внимание на ключевых позициях и существенных местах.

К специальным средствам относятся:

- картографический принцип силуэта,
- средства управления восприятием,
- фиксация начала и конца,
- шапка.

СРЕДСТВА УПРАВЛЕНИЯ ВОСПРИЯТИЕМ

Как привлечь внимание читателя к тем или иным точкам на чертеже алгоритма? Для этого служат средства привлечения внимания, или средства управления восприятием.

Философ Марио Бунге полагает, что интуицию можно, в частности, трактовать как «быстрое восприятие, воображение...» [159]. Интуитивное восприятие осуществляется без слов, без вербализации. Исходя из этого, в языке ДРАКОН используются следующие средства управления восприятием:

- увеличение толщины важных линий,
- выделение шрифтом и кеглем,

- акцент на иконах Время,
- треугольные маркеры,
- ветка-комментарий.

Прелесть в том, что человеку не нужно обдумывать действия при переключении внимания на нужные (выделенные) объекты; они как бы сами бросаются в глаза, сами притягивают взгляд. Это происходит потому, что – благодаря умелому использованию средств управления восприятием – выбранные объекты имеют неоспоримое преимущество перед соседями. Они кажутся более привлекательными и непроизвольно, без участия сознания, попадают в фокус внимания. Перечислим приемы для привлечения внимания на рис. 102.

Основой зрительной композиции являются шампуры. Необходимо выделить шесть шампуров – по одному в каждой ветке. По этой причине все шампуры нарисованы жирными линиями.

Текст в иконах Заголовок и Имя ветки играет важную роль и имеет высший приоритет. Поэтому кегль увеличен и выбран жирный шрифт.

На действия и решения врача накладываются ограничения по времени. Чтобы акцентировать внимание на иконах Время, они выделены желтым цветом.

Два маркера «черный треугольник» позволяют легко заметить веточные циклы.

ГДЕ НАЧАЛО, ГДЕ КОНЕЦ

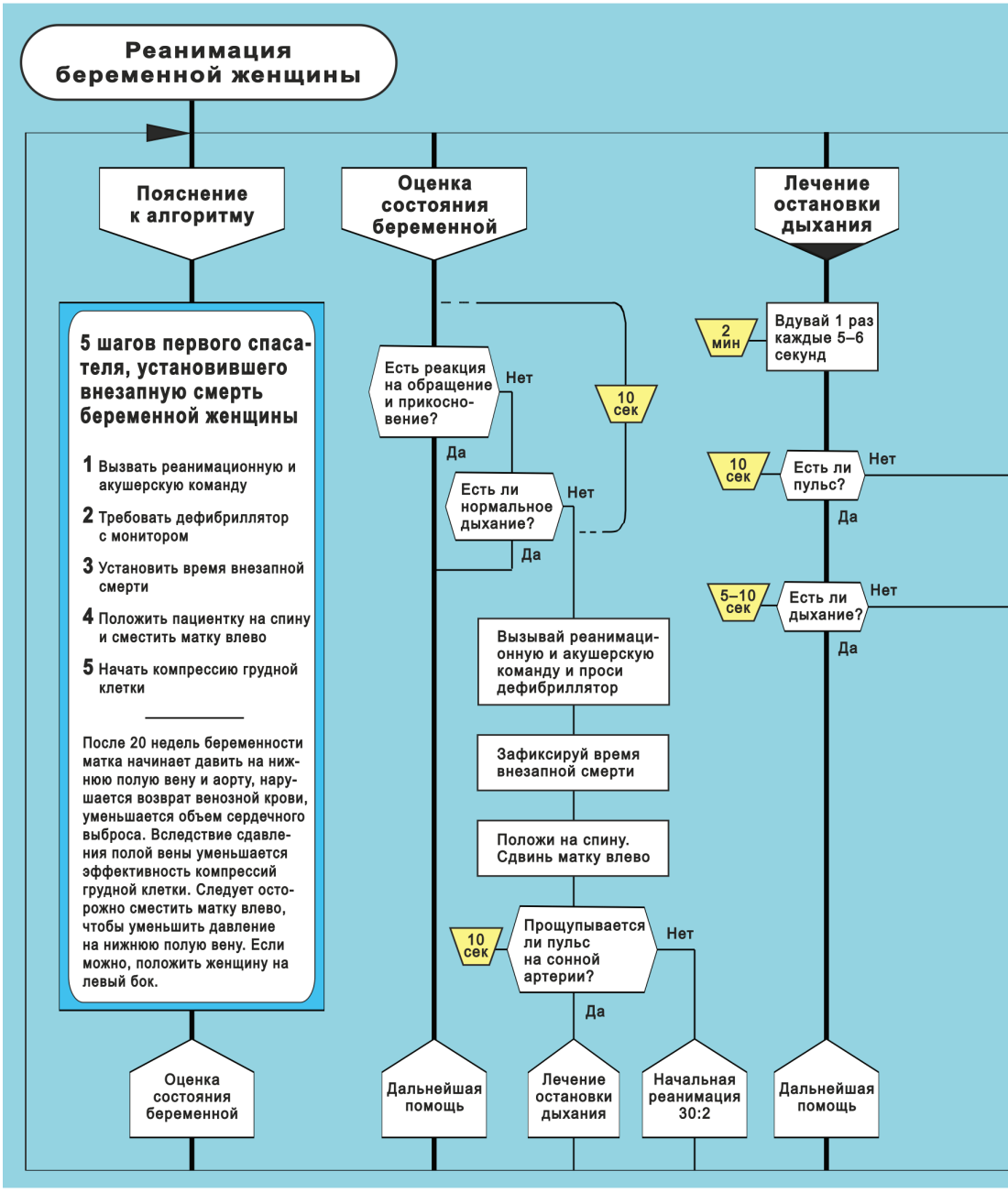
Иконы Заголовок и Конец являются пограничными, они обозначают границы алгоритма во времени и пространстве. Все маршруты выходят из Заголовка и собираются в Конце.

Большую роль играет договоренность о месте размещения этих икон. Они должны всегда находиться на одном и том же, заранее известном, фиксированном месте. В блок-схемах такая договоренность отсутствует, что приводит к хаотическому размещению начала и конца, которые прыгают с места на место.

Язык ДРАКОН имеет на этот счет четкие и неизменные правила. В силуэте Заголовок всегда находится в левом верхнем углу. И только там.

Икона Конец также имеет постоянное место – в правом углу, в конце последней ветки.

Почему необходимо точно фиксировать позицию этих икон? Потому, что это облегчает изучение новых и оживление в памяти забытых алгоритмов. Танцевать надо от печки. Проникновение в алгоритм облегчается, если читатель заранее знает, где следует искать две ключевые фигуры: Заголовок и Конец.


Реанимация беременной женщины
Пояснение к алгоритму
Оценка состояния беременной
Лечение остановки дыхания
5 шагов первого спасателя, установившего внезапную смерть беременной женщины
1 Вызвать реанимационную и акушерскую команду
2 Требовать дефибриллятор с монитором
3 Установить время внезапной смерти
4 Положить пациентку на спину и сместить матку влево
5 Начать компрессию грудной клетки
После 20 недель беременности матка начинает давить на нижнюю полую вену и аорту, нарушается возврат венозной крови, уменьшается объем сердечного выброса. Вследствие сдавления полой вены уменьшается эффективность компрессий грудной клетки. Следует осторожно сместить матку влево, чтобы уменьшить давление на нижнюю полую вену. Если можно, положить женщину на левый бок.
Оценка состояния беременной
Есть реакция на обращение и прикосновение?
Нет
Да
10 сек
Есть ли нормальное дыхание?
Нет
Да
Вызывай реанимационную и акушерскую команду и проси дефибриллятор
Зафиксируй время внезапной смерти
Положи на спину. Сдвинь матку влево
10 сек
Прощупывается ли пульс на сонной артерии?
Нет
Да
Дальнейшая помощь
Лечение остановки дыхания
Начальная реанимация 30:2
2 мин
Вдувай 1 раз каждые 5–6 секунд
10 сек
Есть ли пульс?
Нет
Да
5–10 сек
Есть ли дыхание?
Нет
Да
Дальнейшая помощь

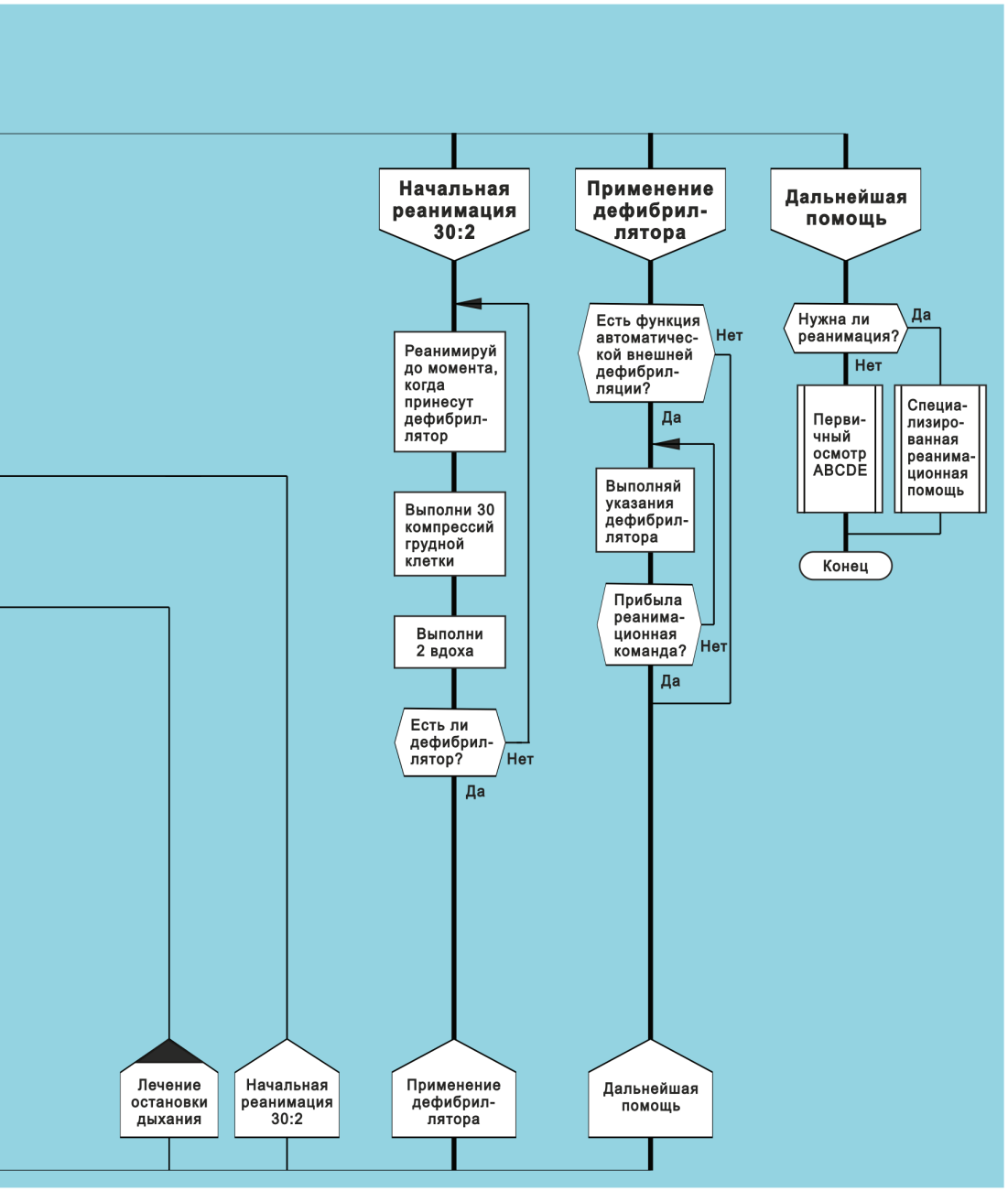


Рис. 102. Алгоритм «Реанимация беременной женщины» [160]

ШАПКА ДЛЯ АЛГОРИТМА РЕАНИМАЦИИ

Шапка – двухэтажная фигура, которая задает смысловой костяк алгоритма (рис. 103). Изюминка в том, что шапка делает человеку своевременные подсказки, настраивающие его на нужный лад и облегчающие работу ума.

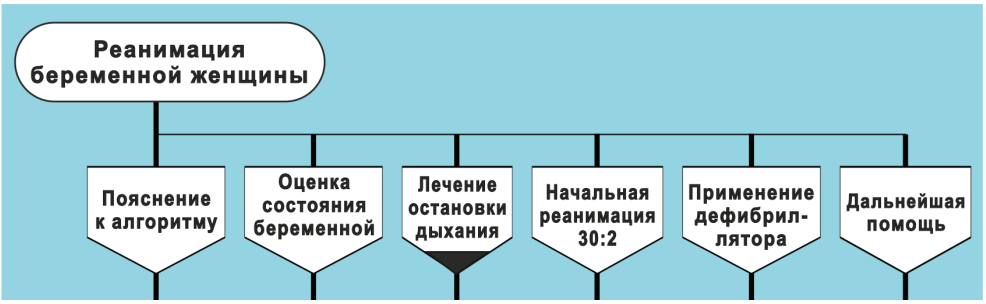


Рис. 103. Шапка сложного алгоритма на рис. 102

Прочитаем подсказки для рис. 102, 103.

- Как называется задача? *(Читаем заголовок алгоритма).*
  «Реанимация беременной женщины».
- Из скольких частей она состоит? *(Считаем иконы «Имя ветки»).*
  Из шести.
- Как называется каждая часть? *(Читаем текст в иконах «Имя ветки»).*
  1. «Пояснение к алгоритму».
  2. «Оценка состояния беременной».
  3. «Лечение остановки дыхания».
  4. «Начальная реанимация 30:2».
  5. «Применение дефибриллятора».
  6. «Дальнейшая помощь».

Сколько времени нужно, чтобы впитать эту информацию? Минуту? Две? Три? Пять?

Шапка дает общее представление, без деталей. Благодаря шапке читатель получает ориентировку о проблеме практически сразу. Причем, что очень важно, время ориентировки почти не зависит от сложности задачи и составляет считанные минуты.

Из шапки мы узнали названия веток. После этого, как всегда, приступаем к двум стандартным операциям:

- изучению содержания веток,
- анализу их взаимодействия.

Чтобы выполнить эти операции и тщательно изучить алгоритм на рис. 102, подготовленному читателю вряд ли потребуется больше, чем

двадцать-тридцать минут. (Речь, разумеется, идет о специалистах или студентах, а не о человеке с улицы).

ИЗУЧАЕМ ВЕТКУ «ПОЯСНЕНИЕ К АЛГОРИТМУ»

На рис. 102 первая ветка «пустая»; она не выполняет никаких действий. Зачем она нужна?

Рассмотрим случай, когда алгоритм используется в медицинском учебнике в качестве иллюстрации. Обычно его сопровождает развернутый пояснительный текст, расположенный на соседних страницах.

Предположим, авторы учебника заботятся об эргономике и желают сделать материал более дружелюбным. В этом случае можно создать дополнительное удобства для читателей. Надо выделить из пояснительного текста наиболее важные мысли и вставить их непосредственно в чертеж алгоритма в качестве краткого введения.

Для этой цели необходимо:

- добавить в алгоритм еще одну ветку, которая называется «ветка-комментарий» (рис. 104);
- поместить в нее большую икону Комментарий, которая занимает всю ветку;
- записать в Комментарии самые ценные мысли.

По этому рецепту создана крайняя левая ветка на рис. 102. В качестве важных мыслей использованы два избранных отрывка:

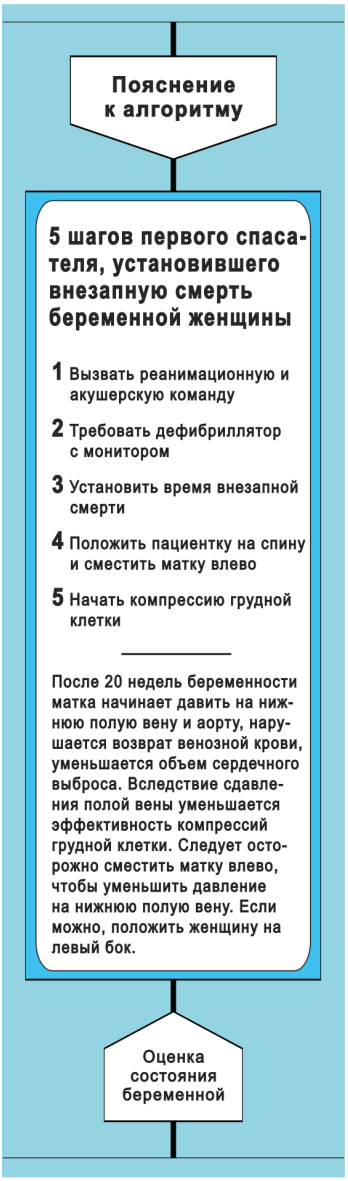


Рис. 104. Ветка-комментарий

- 5 шагов первого спасателя, установившего внезапную смерть беременной женщины [161];
- информация о процессах, протекающих в организме после 20-й недели беременности, которые могут привести к нарушению возврата венозной крови и уменьшению сердечного выброса [162].

Что мы при этом выиграли? Мы поместили наиболее важную информацию рядом с алгоритмом – на одной зрительной сцене. Тем самым мы избавили читателя от необходимости листать страницы туда-сюда, попеременно воспринимая текст и чертеж, находящиеся на разных страницах. Мы создали дополнительные удобства для читателя, обеспечив снижение нагрузки на его оперативную память.

ИЗУЧАЕМ ВЕТКУ «ОЦЕНКА СОСТОЯНИЯ БЕРЕМЕННОЙ»

В конце первой ветки в иконе Адрес читаем «Оценка состояния беременной» – это имя следующей ветки, куда следует перейти.

Данная команда переводит наш взгляд (и бегунок) в начало второй ветки. Для удобства чтения ветка выделена из рис. 102 и представлена на рис. 105.

Ветка «Оценка состояния беременной» существенно отличается от тех, что нам уже знакомы. Новинка в том, что она содержит три иконы Адрес. Три разных Адреса указывают на переход в три разные ветки. Зачем это нужно?

Мы знаем, что надписи в иконах Адрес играют роль суфлера и советчика – они однозначно определяют порядок выполнения веток.

В предыдущих примерах (рис. 87, 94, 96) каждая ветка, кроме последней, имела только одну икону Адрес. Поэтому ветки работали последовательно, соблюдая железный порядок: сначала первая, после первой всегда вторая, после второй всегда третья и т. д.

Однако подобная жесткая дисциплина нужна далеко не всегда. Во многих случаях необходимо, чтобы ветка служила «большой развилкой»

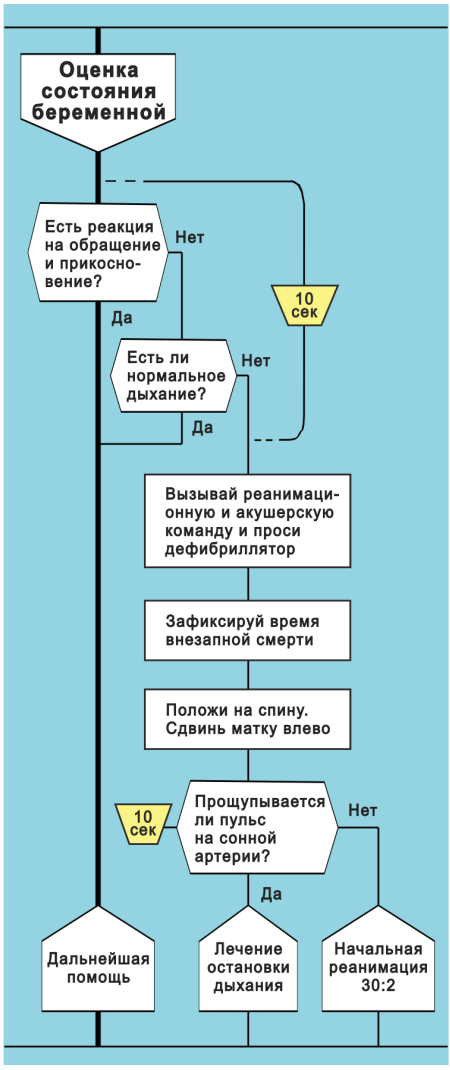


Рис. 105. Ветка «Оценка состояния беременной»

и позволяла выбирать различные пути, т. е. указывала на несколько разных веток.

Ветка «Оценка состояния беременной» как раз и выполняет такую функцию. Имея три иконы Адрес, она дает возможность совершить прыжок в три разные ветки:

- «Дальнейшая помощь».
- «Лечение остановки дыхания».
- «Начальная реанимация 30:2».

ИКОНА «ВРЕМЯ»

Обратите внимание на две маленькие иконы – перевернутые трапеции желтого цвета. Они обозначают Время и выполняют важную функцию – указывают врачу предельное время, отводимое на ту или иную операцию.

Например, верхняя икона «10 секунд» на рис. 105 напоминает, что врач обязан за это время успеть выполнить две операции:

- выяснить, есть ли реакция пациентки на обращение и прикосновение,
- выяснить, есть ли у пациентки нормальное (не агональное) дыхание.

ЛИШНЮЮ ИКОНУ СЛЕДУЕТ УДАЛИТЬ

Рассмотрим ситуацию подробнее. Для этого проанализируем икону Время «10 секунд», соединенную с иконой Вопрос «Прощупывается ли пульс на сонной артерии?» (рис. 105).

Икона Вопрос в данном случае несет двойную нагрузку. Во-первых, она обозначает Действие: «Прощупай пульс на сонной артерии». Во-вторых, по результатам Действия врач должен принять Решение: прощупывается пульс или нет?

Поскольку речь идет о реанимации, время дорого и счет идет на секунды. Желтая икона Время напоминает врачу: на обе операции (выполнение Действия и принятие Решения) ему отводится всего 10 секунд.

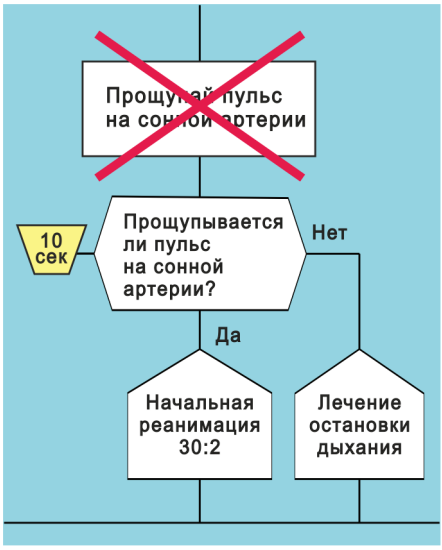


Рис. 106. Икона Действие здесь не нужна

Вопрос. Некоторые люди считают, что любое действие (например, «Прощупай пульс на сонной артерии») следует писать в иконе Действие. Почему в данном случае икона Действие отсутствует на рис. 105?

Ответ. Потому что в ней нет никакой необходимости. Если послушать горе-советчиков и добавить ненужную икону (как показано на рис. 106), то надписи в иконах Действие и Вопрос будут почти полностью совпадать:

- «Прощупай пульс на сонной артерии».
- «Прощупывается ли пульс на сонной артерии?»

При восприятии одинакового текста читатель не получает новой информации и зря теряет время. Это может вызвать ненужную заминку и даже раздражение. По этой причине икону Действие удаляют – на рис. 106 она зачеркнута крестом.

Правило лишней иконы

Если две иконы (Действие и Вопрос) соединены последовательно, причем текст в обеих иконах почти полностью повторяется, икона Действие в медицинском алгоритме является лишней. Ее следует удалить.

КАК ИСПРАВИТЬ ОШИБКУ

Алгоритм «Реанимация беременной женщины» мы заимствовали из источника [160]. При этом выявилась интересная подробность. В оригинале ветка «Оценка состояния беременной» изображена, как показано на рис. 107. Обратите внимание на порядок икон Адрес:

- «Начальная реанимация 30:2».
- «Лечение остановки дыхания».
- «Дальнейшая помощь».

Это значит, что нарушено основополагающее правило «Чем правее, тем хуже». Действительно, самая неблагоприятная для пациента ситуация «Начальная реанимация 30:2» по ошибке оказалась слева. А должна быть справа! Кроме того, икона Адрес «Дальнейшая помощь» должна занимать не правую, а крайнюю левую позицию.

Отсюда вытекает, что мы обнаружили эргономическую неточность (ошибку) в оригинале. Чтобы устранить дефект, необходимо несколько раз осуществить рокировку. Посмотрим, как это делается.

Первый раз выполняем рокировку в развилке «Есть реакция на обращение и прикосновение?». Это позволяет выпрямить главный маршрут и направить его по шампуру. Но этого мало.

Нужна вторая рокировка в развилке «Есть ли нормальное дыхание?». Благоприятный ответ Да мы также подключаем к шампуру.

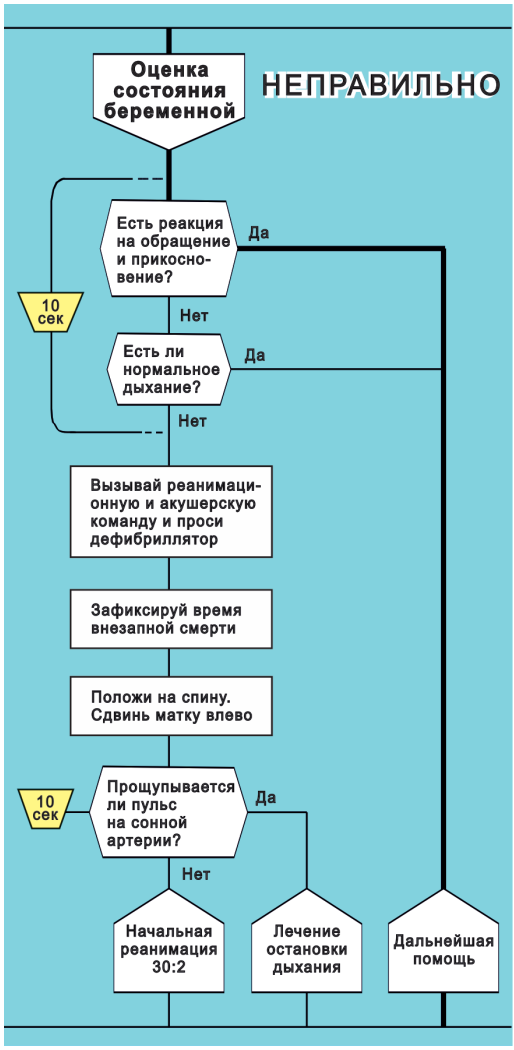


Рис. 107. Ветка с ошибкой. Нарушено правило «Чем правее, тем хуже»

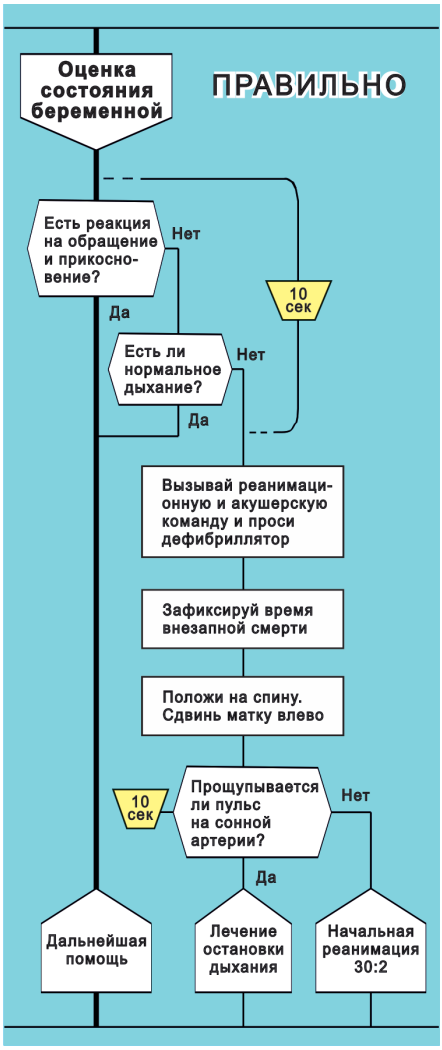


Рис. 108. Ошибка исправлена с помощью рокировки

Третью и последнюю рокировку делаем в развилке «Прослушивается ли пульс на сонной артерии?». Благодаря этому икона «Начальная реанимация 30:2» перемещается туда, куда надо – в крайнее правое положение.

В результате трех рокировок мы значительно улучшили схему. Но не полностью. Возникла неожиданная неприятность с верхней иконой «10 секунд». Если сохранить ее на прежнем месте (слева от ветки), нижний конец будет пересекать шампур. Это недопустимо, потому что пересечения категорически запрещены. Как же быть?

Выход есть! Надо передвинуть икону «10 секунд» и расположить ее справа от ветки, как показано на рис. 108. И пересечение сразу исчезнет. Такая операция называется *зеркальным отражением* иконы «Время группы» (отражение производится относительно вертикали).

Таким образом, мы решили все трудности. В результате трех рокировок и зеркального отражения удалось устранить выявленные в первоисточнике нарушения правил языка ДРАКОН. Результат показан на рис. 108.

В заключение добавим, что два алгоритма на рис. 107 и 108 являются равносильными. Они отличаются лишь графически.

ИЗУЧАЕМ ВЕТКУ «ЛЕЧЕНИЕ ОСТАНОВКИ ДЫХАНИЯ»

Ветка выделена из рис. 102 и показана на рис. 109. Как и ветка на рис. 105, она служит «большой развилкой» на три направления. И потому имеет три иконы Адрес.

Икона «2 минуты» прицеплена к иконе Действие «Вдувай 1 раз каждые 5–6 секунд». Смысл в том, что врач должен делать пациентке искусственные вдохи (вдувая в нее кислород каждые 5–6 секунд) в течение двух минут.

Нижние две иконы Время связаны с иконами Вопрос. Они указывают предельное время для выполнения операций. На принятие решения «Есть ли пульс?» врачу отводится 10 секунд. Чтобы выяснить «Есть ли дыхание?» дается 5–10 секунд.

Согласно *Правилу лишней иконы* перед иконами Вопрос отсутствуют иконы Действие с повторяющейся информацией.

Таким образом, икона Вопрос «Есть ли пульс?» имеет двойную функцию:

- Во-первых, врач реализует Действие «Нащупай пульс»,
- Во-первых, врач принимает Решение «Есть ли пульс?» с ответом Да или Нет.

Икона «Есть ли дыхание?» также реализует двойную функцию.

ВЕТОЧНЫЙ ЦИКЛ

В главе 9 мы уже познакомились с *простым циклом*. Простой цикл легко узнать по характерной горизонтальной стрелке, направленной влево. Петля простого цикла всегда закручена против часовой стрелки (см. рис. 79, 81, 83).

Существует иной тип цикла, для которого стрелка не нужна. Он образуется с помощью иконы Адрес. Каким образом?

Если икона Адрес указывает на свою собственную (или на более левую) ветку, образуется *повторение* действий, то есть цикл.

Пример показан на рис. 109, где в иконе Адрес написано имя своей родной ветки «Лечение остановки дыхания».

Представим себе, что при выполнении этой ветки бегунок спускается вниз, попадает в икону Вопрос «Есть ли дыхание?», и врач видит, что дыхания нет. Это значит, что бегунок выходит через Нет, проходит через икону Адрес «Лечение остановки дыхания» и снова возвращается в начало той же самой ветки.

Если не изменятся условия в обеих иконах Вопрос, бегунок будет снова и снова утюжить ветку «Лечение остановки дыхания». Это и есть *веточный цикл*. Он обозначается двумя черными треугольниками (рис. 109).

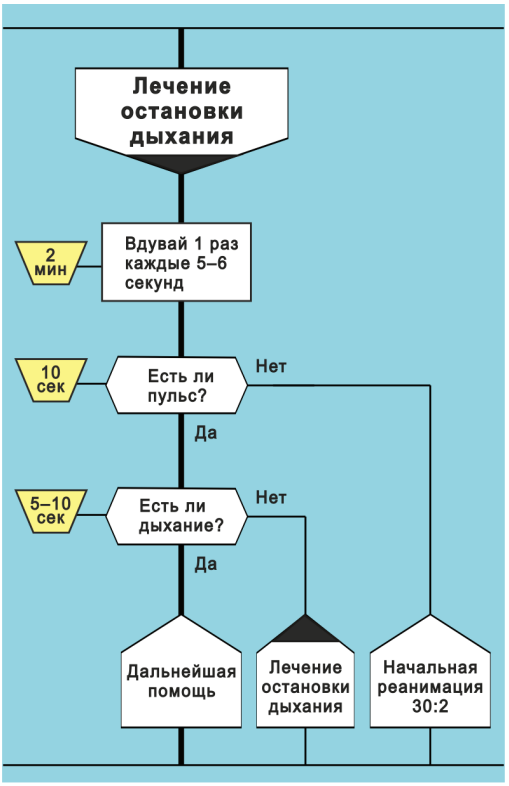


Рис. 109. Ветка «Лечение остановки дыхания»

Когда цикл закончит работу? Возможны два варианта:

1. Если у пациентки пропал пульс, бегунок покидает икону «Есть ли пульс?» через Нет и далее через икону Адрес прыгает в ветку «Начальная реанимация 30:2».
2. Если же, к счастью, у пациентки появилось дыхание, бегунок выходит из иконы «Есть ли дыхание?» через Да и затем через Адрес перемещается в ветку «Дальнейшая помощь».

Какова роль черных треугольников? Для работы алгоритма они совершенно не нужны. Они ни на что не влияют. Если их убрать, ничего не изменится.

Зачем же они поставлены? Только для того, чтобы служить удобным зрительным ориентиром.

Если бы треугольников не было, пришлось бы долго разбираться, чтобы убедиться в том, что в одной из икон Адрес записан адрес собственной ветки. А треугольники сразу бросаются в глаза и экономят время.

Может случиться так, что в алгоритме потребуются несколько веточных циклов. В таком случае, чтобы не запутаться, нужно использовать разные маркеры-треугольники, представленные в справочнике на рис. 22.

Что такое веточный цикл?

- Это цикл, при котором в иконе Адрес написано имя своей собственной (или более левой) ветки.
- Чтобы легко найти веточный цикл, его помечают двумя черными треугольниками.

ИЗУЧАЕМ ВЕТКУ «НАЧАЛЬНАЯ РЕАНИМАЦИЯ 30:2»

В эту ветку можно попасть двумя способами:

- Из ветки «Оценка состояния беременной», когда в иконе «Прощупывается ли пульс на сонной артерии?» получен ответ Нет.
- Из ветки «Лечение остановки дыхания», когда из иконы «Есть ли пульс?» выходим через Нет.

Формула 30:2 означает, что спасатель поочередно выполняет:

- 30 компрессий грудной клетки,
- 2 вдоха в легкие пациентки.

На рис. 110 регулярное повторение этих действий изображено в виде цикла. Цикл заканчивается, когда принесут дефибриллятор.

Описание ветки «Применение дефибриллятора» мы опускаем, так как все используемые в ней иконы нам уже известны.

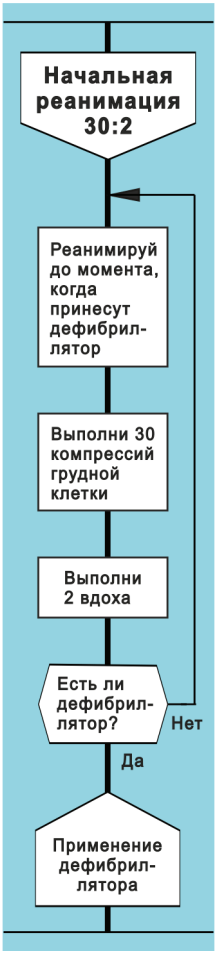


Рис. 110. Ветка «Начальная реанимация 30:2»

ГЛАВНЫЙ МАРШРУТ СИЛУЭТА И ПРАВИЛО ВЕЗЕНИЯ

Предположим, что пациент является «везунчиком», в жизни ему всегда везет. Можно сказать, что жизнь такого пациента подчиняется «Правилу везения».

Заодно вспомним, что главный маршрут алгоритма – это наиболее благоприятный для пациента путь, ведущий из Заголовка в Конец. Слова «наиболее благоприятный маршрут» и «Правило везения» имеют один и тот же смысл.

Поясним сказанное применительно к алгоритму на рис. 102.

Чтобы построить главный маршрут силуэта, следует держаться Правила везения – во всех развилках надо выбирать вариант, который является наилучшим для нашей пациентки, беременной женщины.

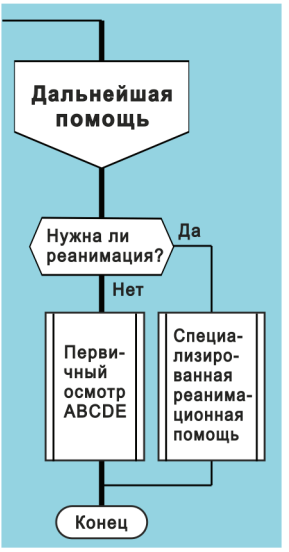


Рис. 111. Ветка «Дальнейшая помощь»

Итак, приступим. Из Заголовка на рис. 102 путь идет вниз по шампуру первой ветки. Через икону Адрес бегунок попадает на шампур второй ветки. Здесь появляется вопрос: «Есть ли реакция на обращение и прикосновение?». Руководствуясь Правилом везения, бегунок отбрасывает плохой ответ и выбирает хороший, согласно которому пациентка успешно реагирует на прикосновение и обращение. Бегунок скользит вниз по шампуру второй ветки и опускается в икону Адрес «Дальнейшая помощь».

Таким образом, бегунок пропускает три ветки на рис. 102, описывающие экстренную помощь:

- «Лечение остановки дыхания».
- «Начальная реанимация 30:2».
- «Применение дефибриллятора».

В результате такого маневра бегунок обходит неприятности и сразу попадает на вход ветки «Дальнейшая помощь» (рис. 111). Здесь возникает второй и последний вопрос: «Нужна ли реанимация?»

Правило везения требует выбрать ответ Нет. И бегунок по шампуру прямиком спускается до Конца.

Мы убедились, что главный маршрут силуэта на рис. 102 проходит через три шампура (через первую, вторую и последнюю ветки). Это и есть наиболее благоприятный маршрут для нашей беременной женщины.

НЕЯСНОСТЬ НЕОБХОДИМО УСТРАНИТЬ

Внимательный анализ показывает, что главный маршрут алгоритма на рис. 102 содержит неясные места и порождает вопросы. Алгоритм посвящен реанимации, но главный маршрут по какой-то причине обходит все реанимационные действия.

В самом деле, можно ли, исследуя главный маршрут, говорить о том, что произошла внезапная смерть пациентки? Нет, нельзя. Можно ли считать, что у нее пропал пульс? Тоже нет. Наконец, были ли у нее трудности с дыханием? Опять-таки нет. Она нормально реагировала на обращение и прикосновение.

Это странно. Главный маршрут алгоритма не содержит нужных сведений и требует обдумывания. По какой причине у врача возникли подозрения, что наша героиня попала в критическую ситуацию и, возможно, нуждается в реанимации?

Главный маршрут, к сожалению, не позволяет уточнить ситуацию и дать однозначный ответ. Приходится констатировать, что в данном алгоритме имеет место неясность. Неясность, которую необходимо устранить.

Читатель может рассматривать описанную ситуацию как домашнее задание и по своему усмотрению устранить недостаток.

АЛГОРИТМ ВЫСОКОЙ ТОЧНОСТИ И НОВАЯ КУЛЬТУРА КЛИНИЧЕСКОГО МЫШЛЕНИЯ

ДРАКОН – алгоритмический язык высокой точности. Он представляет собой принципиально новый инструмент. Раньше таких инструментов просто не существовало.

ДРАКОН обеспечивает высочайшую точность при разработке медицинских алгоритмов. Это обнажает ряд проблем, которые раньше казались незаметными.

В медицине существуют определенные традиции, устоявшиеся привычки, традиционный стиль при описании медицинских алгоритмов. Для этого стиля характерен творческий, неформальный подход. Творческому стилю присуща некоторая размашистость, приблизительность, неточность и необязательность.

Творческий человек сосредоточен на главном, и эту главную идею он тщательно шлифует, совершенствует, уточняет, доводит до блеска и до победного конца.

Однако от главной идеи обычно ответвляются многочисленные боковые маршруты, за которыми очень трудно уследить.

Здесь требуются совсем другие качества: пунктуальность, дотошность, аккуратность, алгоритмическое буквоедство и алгоритмическая педантичность.

Проблема состоит в том, что до сих пор медицина не знала требования о необходимости **полного анализа всех без исключения маршрутов алгоритма**.

Данное требование является принципиально новым. Оно знаменует переход к ***новой культуре клинического мышления***.

СВЕРТКА ИНФОРМАЦИИ ИЛИ ВЫСОКАЯ ТОЧНОСТЬ?

Существуют две точки зрения на медицинский алгоритм:

- алгоритм как свертка информации,
- алгоритм высокой точности.

Понимание алгоритма как свертки информации получило широкое распространение в медицине. По мнению профессоров Александра Краснова и Игоря Давыдкина, медицинский алгоритм предназначен для сворачивания информации. Они пишут:

> «За последние годы вышло немало учебных пособий, где предприняты попытки написания профессиональных [медицинских] алгоритмов. Однако чаще всего эти алгоритмы написаны клиницистами в произвольном стиле, не имеют унифицированной общепринятой символики, перегружены числом элементов (счет которых иногда идет на десятки на странице. Поэтому они в принципе не могут быть восприняты как свертка информации (для чего, собственно, и предназначались)» [163, 164].

Так ли это? Можно ли согласиться с тем, что алгоритмы предназначались для свертки информации?

Свертка (сворачивание) информации означает сжатый комплекс информационных сведений, из которого удалено все лишнее и малозначительное, оставлена только самая суть. Излишнее сокращение делает информацию непонятной, при недостаточном сокращении за мелочами теряется суть вопроса [165].

По нашему мнению, трактовка алгоритма как свертки устарела и не имеет будущего. Она не позволяет осуществлять детальный анализ всех без исключения маршрутов алгоритма и является источником ошибок. На смену этой концепции приходит принципиально новая идея – идея медицинского алгоритма высокой точности.

ЭРГОНОМИЧНЫЙ АЛГОРИТМ

Эргономичный алгоритм – алгоритм, удовлетворяющий критерию сверхвысокого понимания. Преимущество эргономичных алгоритмов в том, что они намного понятнее, яснее, нагляднее и доходчивее, чем обычные.

Если алгоритм непонятный, в нем трудно или даже невозможно заметить затаившуюся ошибку.

И наоборот, чем понятнее алгоритм, тем легче найти дефект. Поэтому более понятный, эргономичный алгоритм намного лучше обычного. Лучше в том смысле, что он облегчает выявление ошибок, а это очень важно.

Ведь чем больше ошибок удастся обнаружить при проверке за столом, тем больше вероятность, что вновь созданный алгоритм окажется правильным, безошибочным, надежным. Кроме того, эргономичные алгоритмы удобны для изучения, их проще объяснить другому человеку.

Все в алгоритме понятно и ясно,
Если он сделан эргономично.
Эргономично – это прекрасно!
Эргономично – значит отлично!

КНИЖНЫЙ РАЗВОРОТ

При изложении материала мы постепенно увеличивали сложность и объем алгоритмов. Самые крупные примеры представлены на рис. 96 и 102. Каждый из них занимает две страницы, образующие книжный разворот.

Возникает вопрос: как изображать большие алгоритмы в медицинских учебниках? Можно дать следующие рекомендации.

- Большие алгоритмы лучше размещать в виде силуэта на книжных разворотах.
- Если одного разворота мало, можно продолжить алгоритм-силуэт вправо по горизонтали и занять два (или даже три) соседних разворота.
- Сочленение двух разворотов осуществляется с помощью двух икон Соединитель, как показано на рис. 112 (по два соединителя с каждой стороны).
- Для книги, содержащей медицинские алгоритмы, желательно использовать формат 70×100 1/16.
- Не следует использовать слишком мелкий шрифт внутри икон; кегль (размер буквы по вертикали) должен быть не меньше 6 или 7 типографских пунктов.

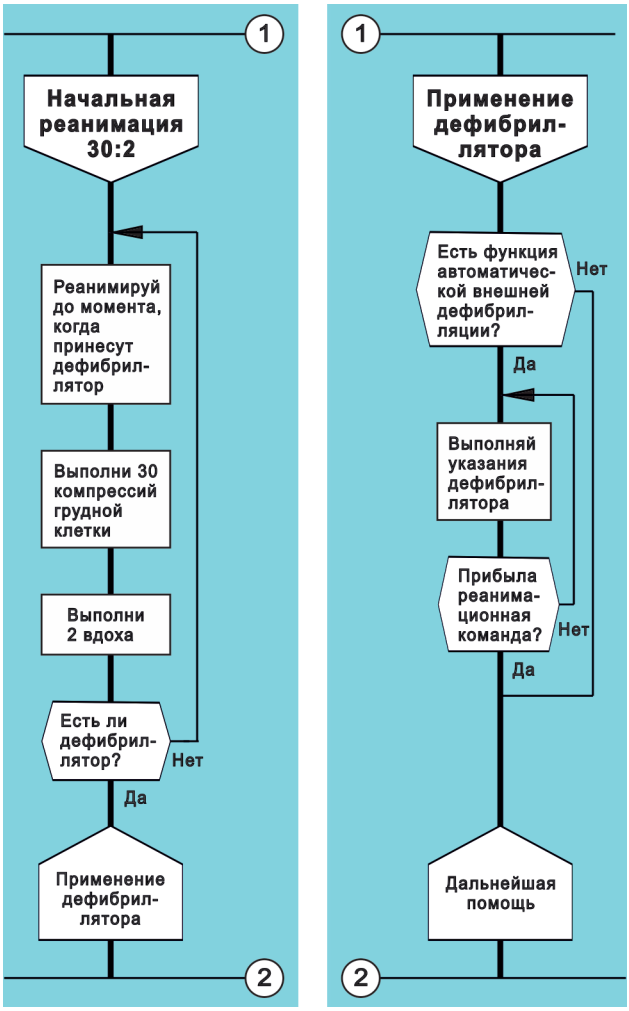


Рис. 112. Переход с листа на лист производится с помощью двух икон Соединитель

ВЫВОДЫ

1. Многоадресный силуэт – это силуэт, содержащий многоадресные ветки.
2. Многоадресный силуэт позволяет:
   - изменять последовательность выполнения веток,
   - выборочно запускать ветки, пропуская некоторые из них,
   - создавать веточные циклы.

3. В языке ДРАКОН используются эффективные средства управления восприятием:
   - картографический принцип силуэта,
   - фиксация начала и конца,
   - шапка,
   - увеличение толщины шампуров,
   - выделение шрифтом и кеглем,
   - акцент на иконах Время,
   - треугольные маркеры,
   - ветка-комментарий.
4. Ветка-комментарий позволяет выделить из пояснительного текста наиболее важные сведения и вставить их непосредственно в чертеж алгоритма.
5. Иконы Время указывают врачу предельное время, отводимое на ту или иную операцию.
6. Если две иконы (Действие и Вопрос) соединены последовательно, причем текст в обеих иконах почти полностью совпадает, икона Действие в медицинском алгоритме является лишней и подлежит удалению.
7. Веточный цикл образуется в случае, если икона Адрес указывает на свою собственную или на более левую ветку.
8. Понимание алгоритма как свертки информации устарело. Будущее принадлежит медицинским алгоритмам высокой точности.
9. Алгоритмы высокой точности нацелены на решение принципиально новой задачи – описание всех без исключения маршрутов алгоритма.