

Визуальный язык ДРАКОН и неклассическая теория алгоритмов

Владимир Паронджанов

17 мая 2008 года

ЖИРОГРАФ И ДРАКОН ПИЛЮГИНА

Жирограф и ДРАКОН Пилюгина



Дружелюбный
Русский
Алгоритмический язык
Который
Обеспечивает
Наглядность

Космонавтика

Кадр из
фильма к
100-летию
Пилюгина

1. Теория формальных языков
2. Теория автоматов
3. Теория вычислимости
4. Теория сложности вычислений

Theory of computation

The field is divided into three major branches: automata theory and language, computability theory, and computational complexity theory, which are linked by the question:
"What are the fundamental capabilities and limitations of computers?"

Ц Е Л Ь

- Облегчить и ускорить понимание алгоритмов
Ч Е Л О В Е К О М
- Исключить ошибки
в алгоритмах
- Заменить стандарт на блок-схемы ГОСТ 19.701-90 и ISO 5807-85 на стандарт языка ДРАКОН



Application Development Without Programmers

**Джеймс Мартин (1933—2013)
автор идеи «Программирование
без программистов»**

Новый научный подход

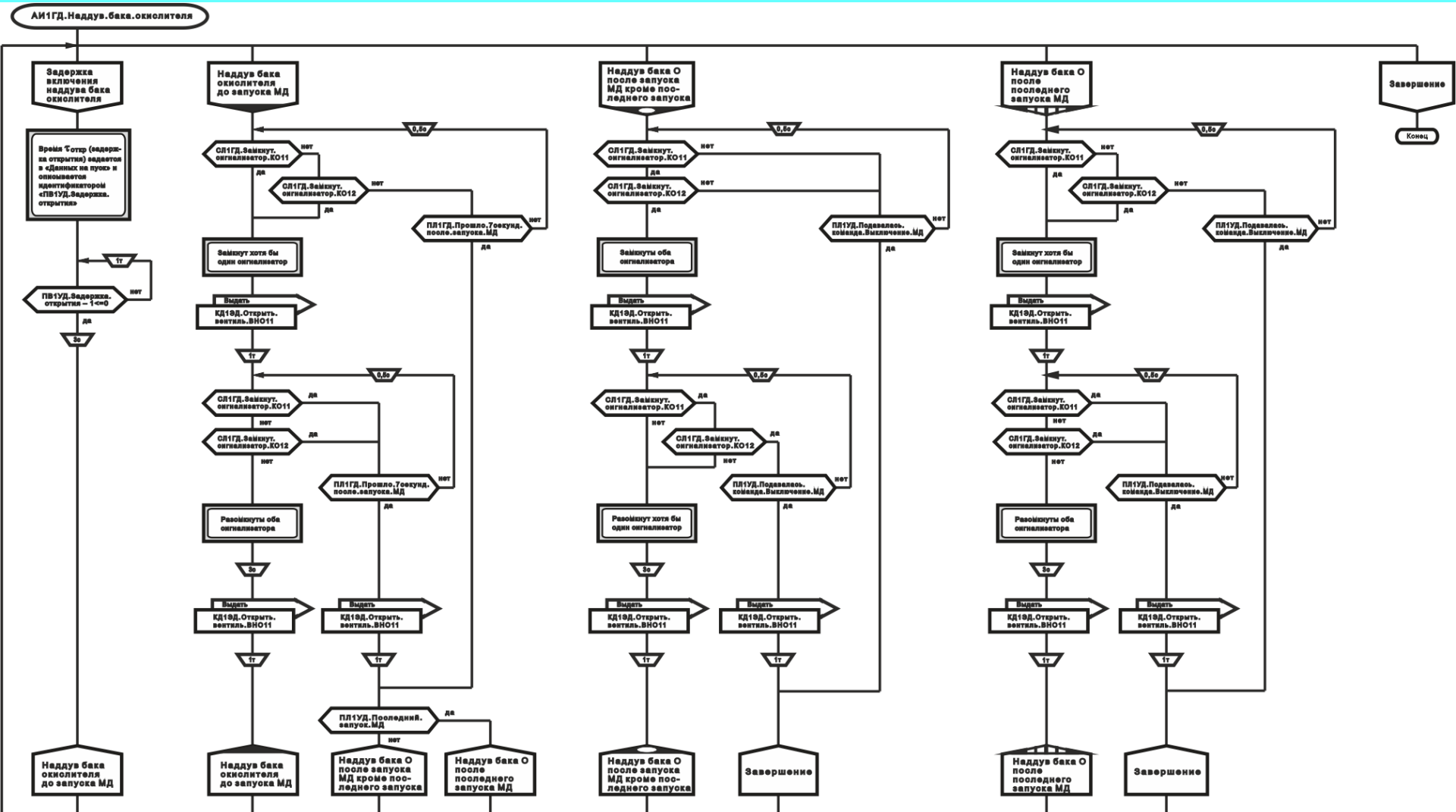
- Объединить АЛГОРИТМЫ и ЭРГОНОМИКУ
- Сделать алгоритмы эргономичными

GUI, WYSIWYG это не то

Ключевые слова для управления вычисли- тельным процессом

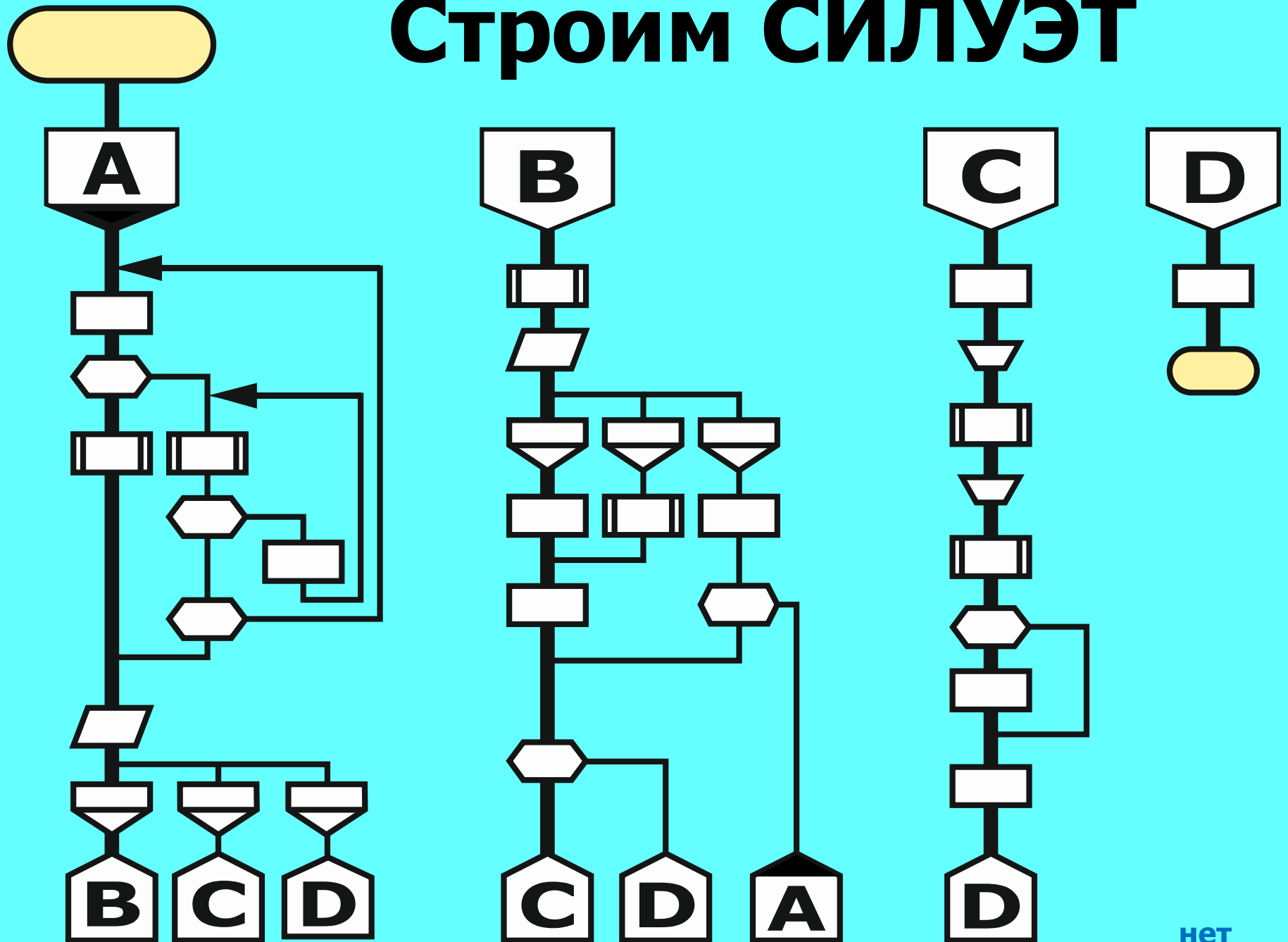
**if, then, else, case, of,
switch, while, do, repeat,
until, for, foreach, loop,
exit, when, last, goto,
break, continue и т.д.**

АИ1ГД.Наддув.бака.окислителя

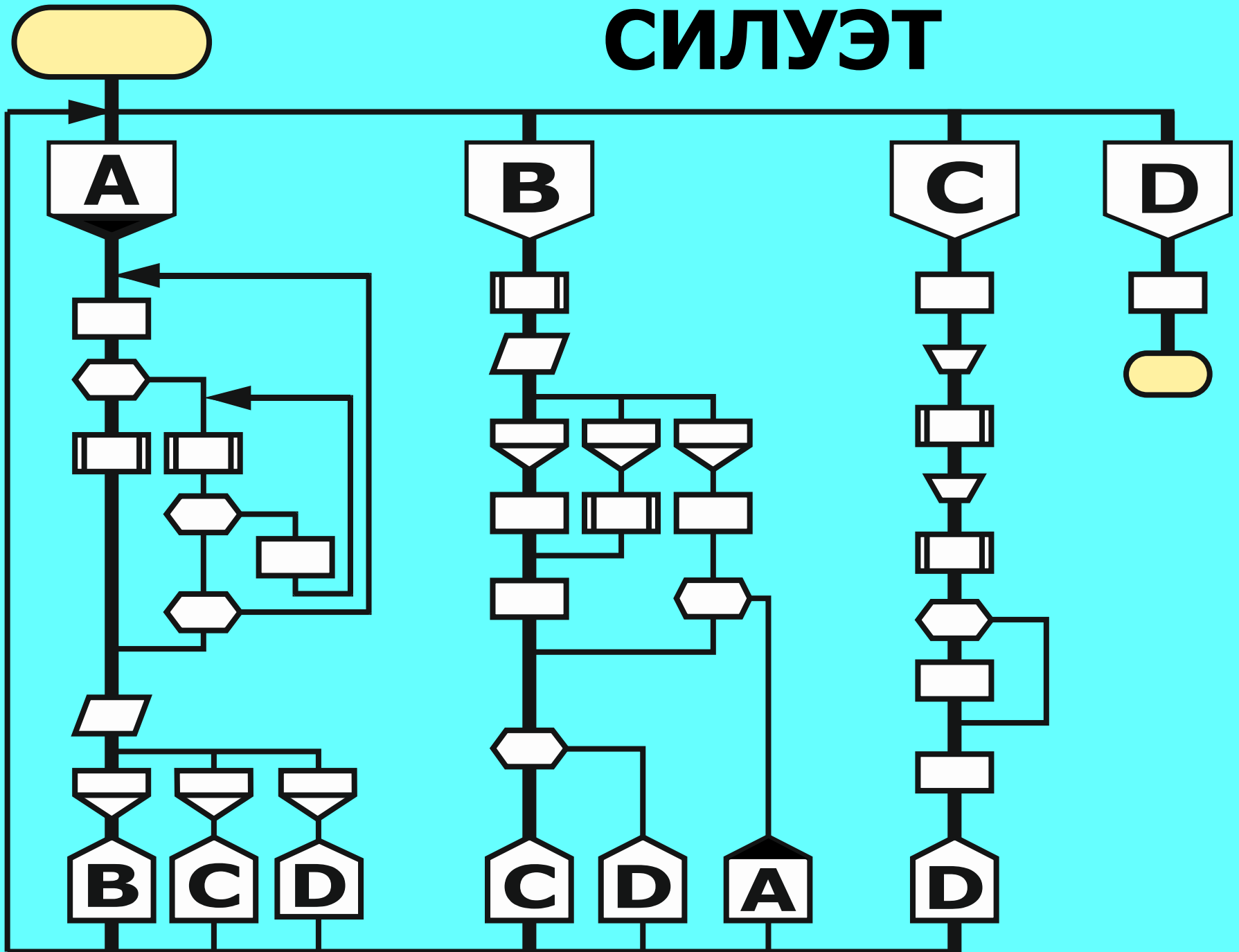


Алгоритмическая конструкция «Силуэт»

Строим СИЛУЭТ



СИЛУЭТ

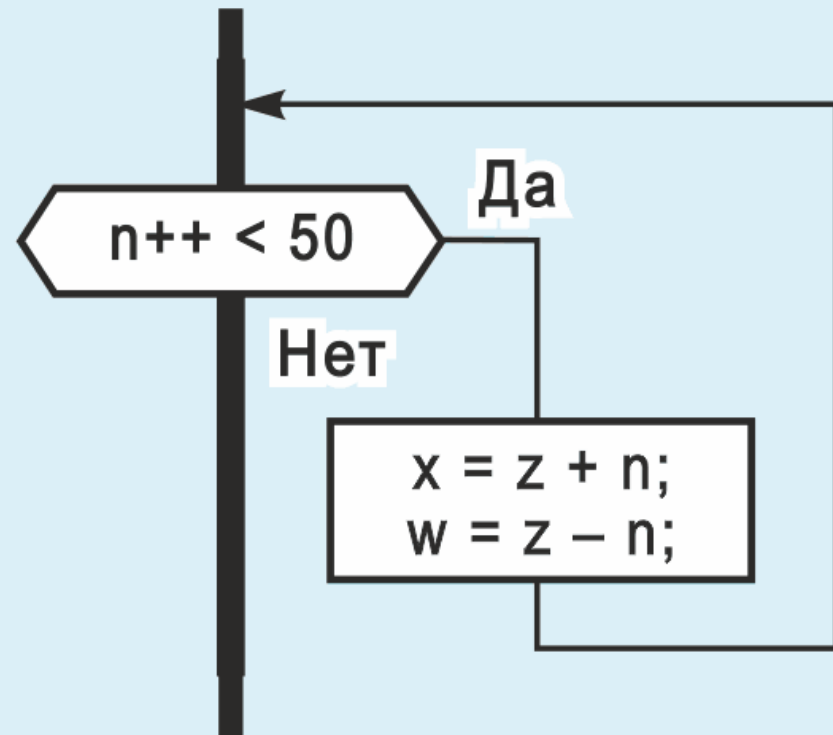


Как удалить Оператор while

Программа на языке Си

```
while ( n++ < 50 )  
{  
    x = z + n;  
    w = z - n;  
} /* while */
```

Программа на языке Дракон-Си

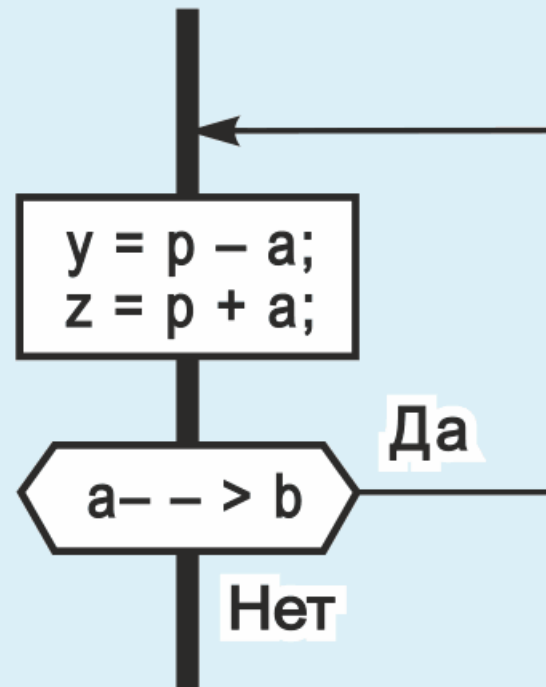


Как удалить Оператор do while

Программа на языке Си

```
do
{
    y = p - a;
    z = p + a;
} while ( a-- > b );
/* do while */
```

Программа на языке Дракон-Си

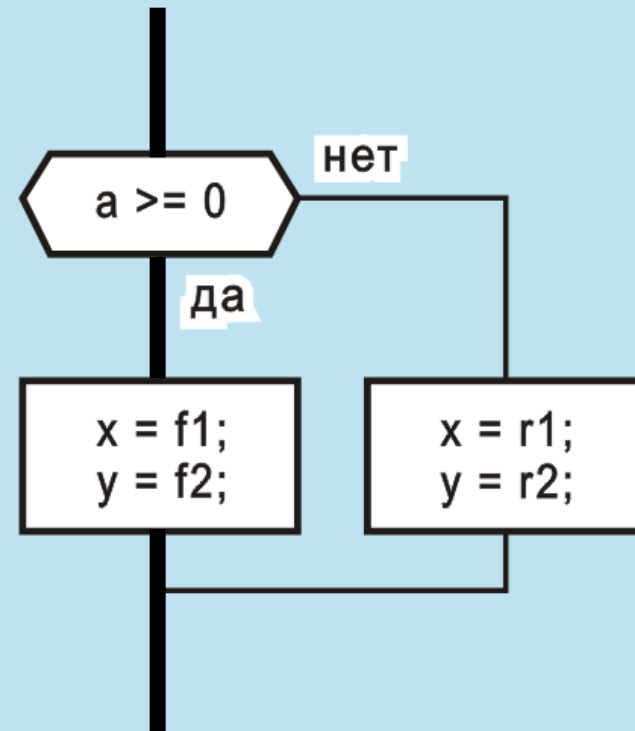


Как удалить Оператор if else

Программа на языке Си

```
if ( a >= 0 )  
{  
    x = f1;  
    y = f2;  
}  
else  
{  
    x = r1;  
    y = r2;  
}
```

Программа на языке Дракон-Си

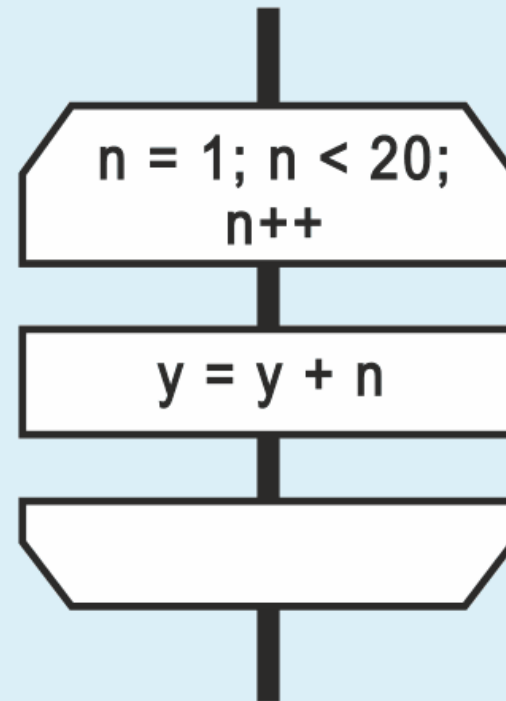


Как удалить Оператор for

Программа
на языке Си

```
for (n=1; n<20; n++)  
    y = y + n;
```

Программа
на языке Дракон-Си



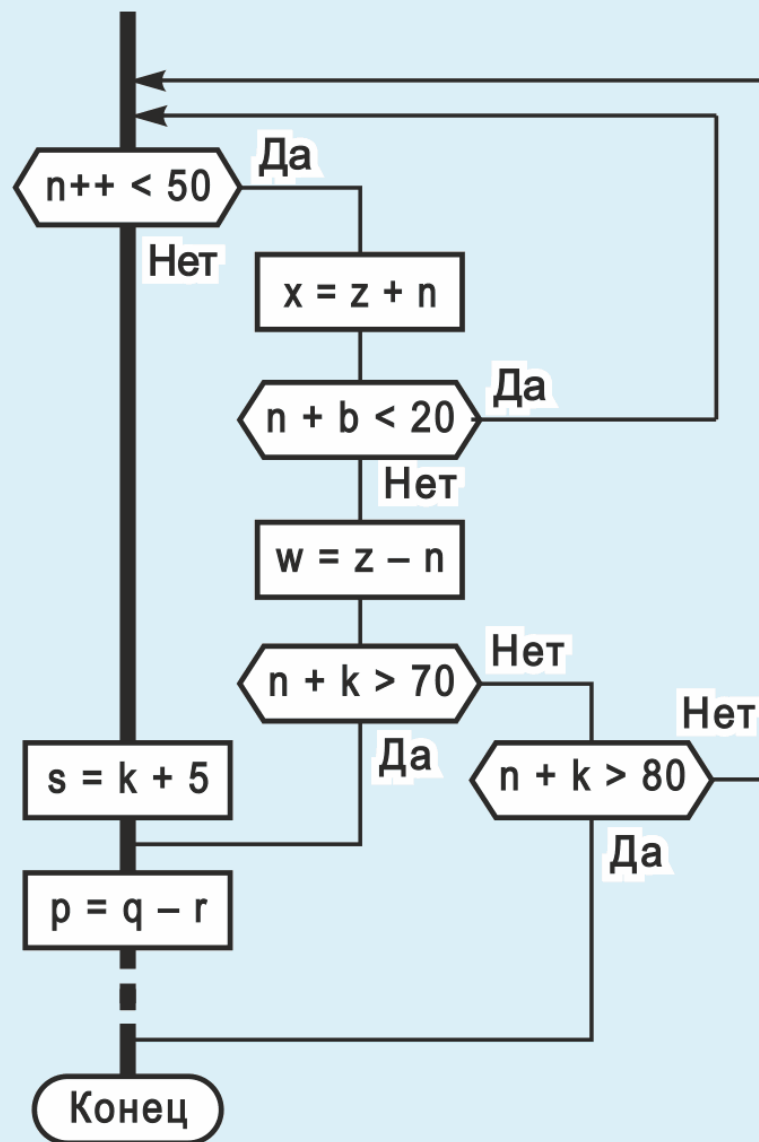
**Как
удалить
Оператор
continue**

**Оператор
return**

**Программа
на языке Си**

```
while (n++ < 50)
{
    x = z + n;
    if (n + b < 20)
        continue;
    w = z - n;
    if (n + k > 70)
        go to m1;
    if (n + k > 80)
        return;
} /* while */
s = k + 5;
m1: p = q - r;
.....
} /* end */
```

**Программа
на языке Дракон-Си**



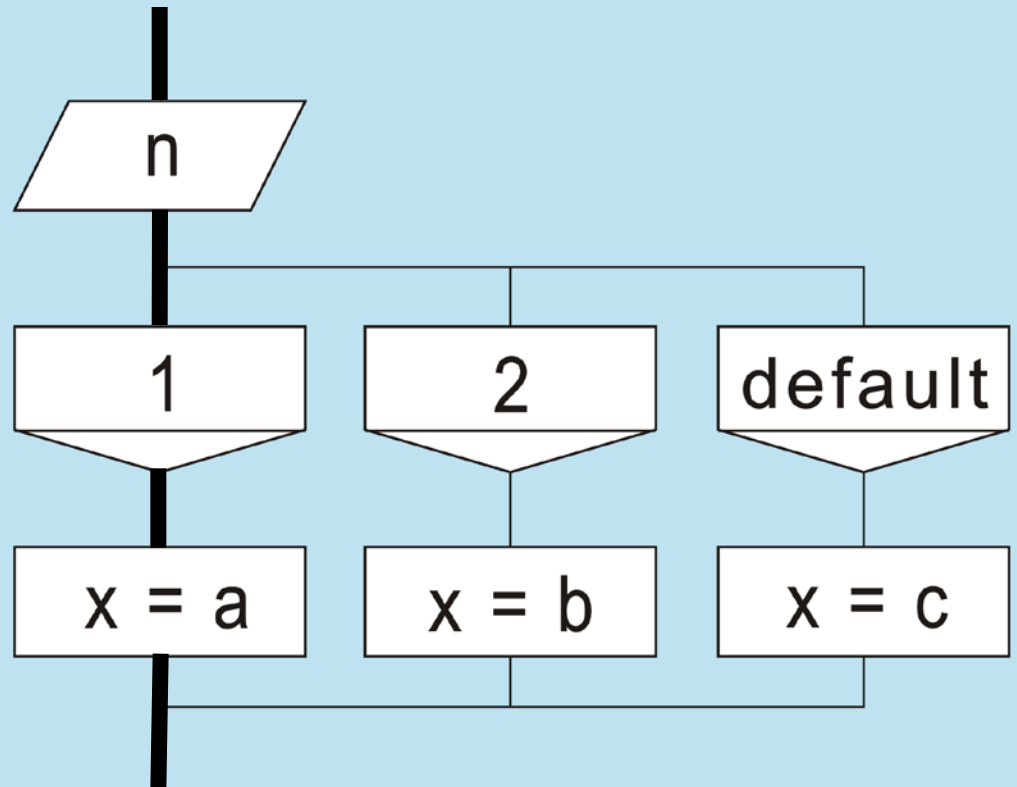
Как удалить

Операторы switch case break

Программа на языке Си

```
switch (n)
{
    case 1:
        x = a;
        break;
    case 2:
        x = b;
        break;
    default:
        x = c;
} /* switch */
```

Программа на языке Дракон-Си



Ненужные элементы

- **switch (два раза),**
- **case 1,**
- **case 2,**
- **break (два раза),**
- **две фигурные скобки,**
- **две круглые скобки,**
- **две косые скобки,**
- **три двоеточия,**
- **пять точек с запятой,**
- **две звездочки.**

Преимущества дракон-программы

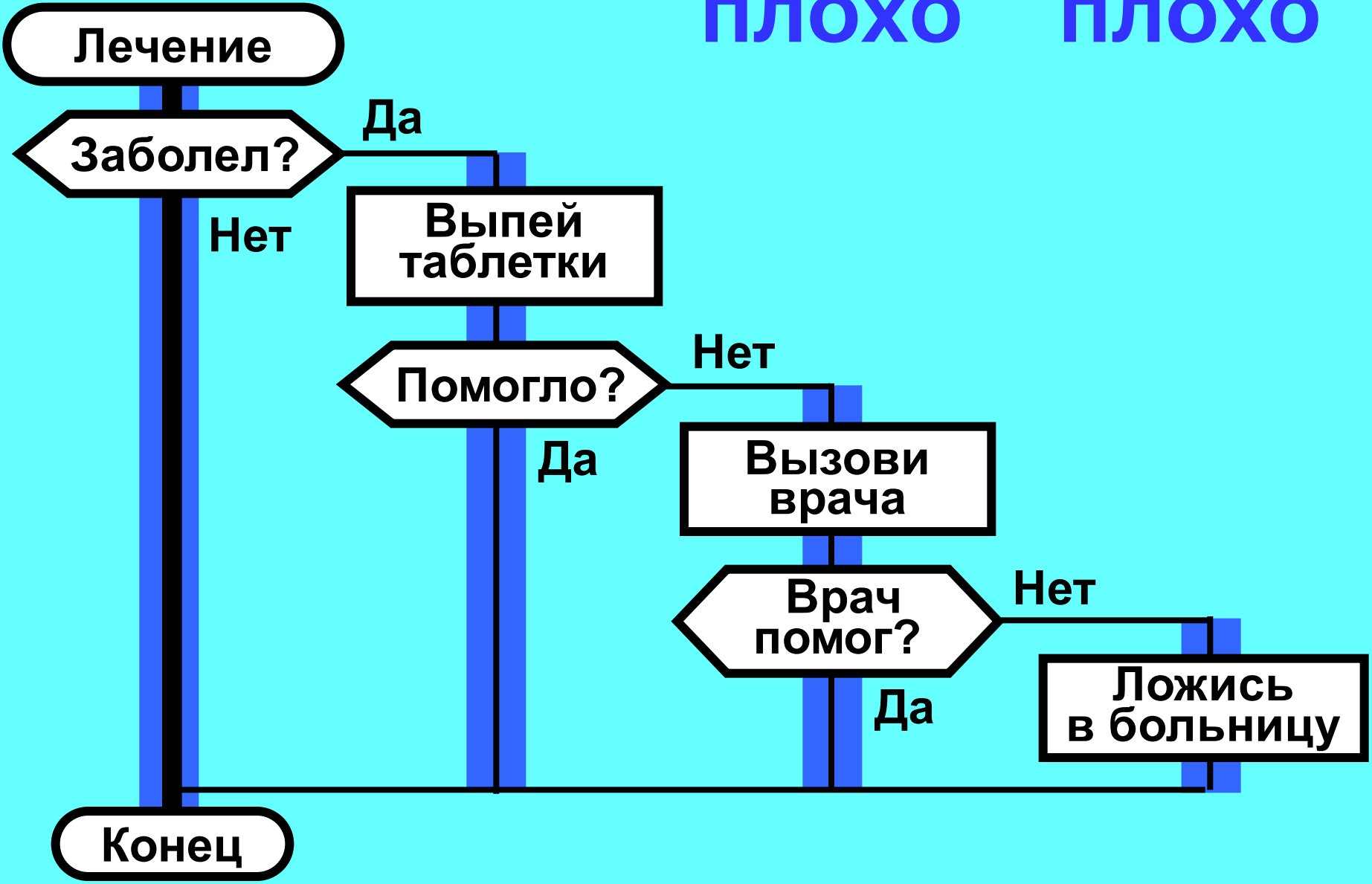
- В дракон-программе число символов сокращается более, чем в три раза, по сравнению с си-программой.
- Остальные символы рассматриваются как мусор и исчезают. Вместо «паразитных» символов используются линии формального чертежа.
- Чертеж гораздо лучше, чем текст, показывает маршруты алгоритма.

Когнитивная эргономика

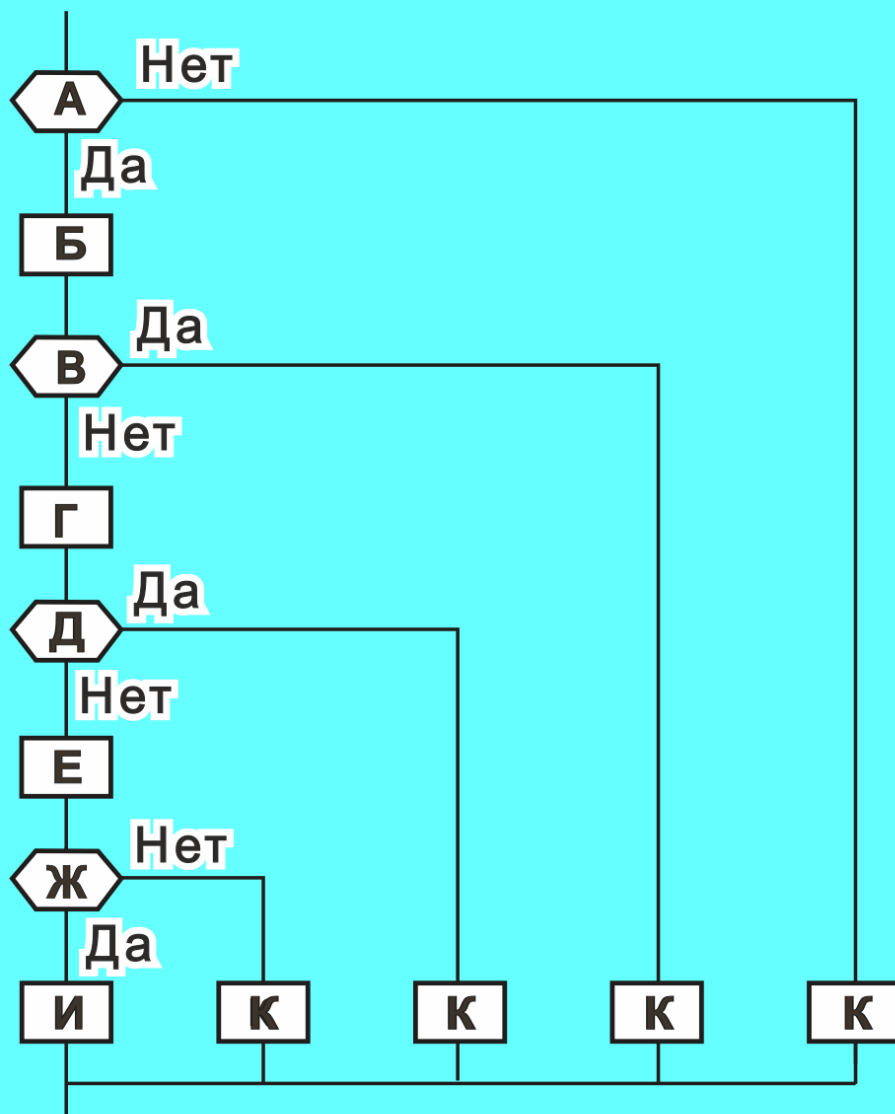
Эргономика интеллекта

Когнитивно-эргономический анализ алгебры
Диофанта Александрийского III век

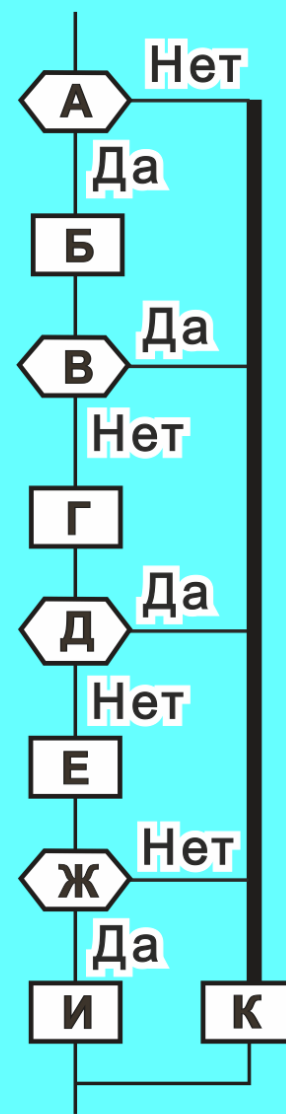
ХОРОШО	ПЛОХО	ОЧЕНЬ ПЛОХО	СОВСЕМ ПЛОХО
Решение			



НЕПРАВИЛЬНО



ПРАВИЛЬНО



Язык ДРАКОН имеет два синтаксиса:

- **графический синтаксис**
- **текстовый синтаксис**

SDL Specification and Description Language

Визуальная ЛОГИКА

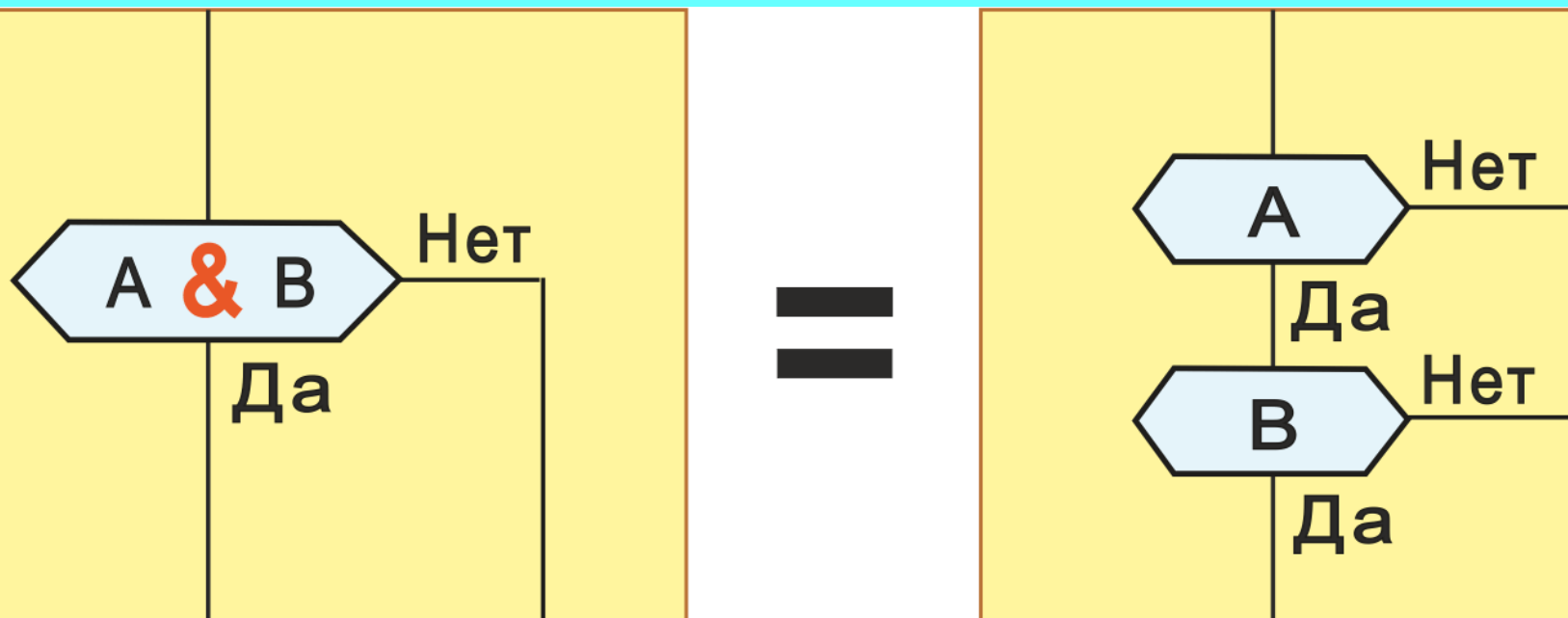
Использование символов алгебры логики

& V ¬

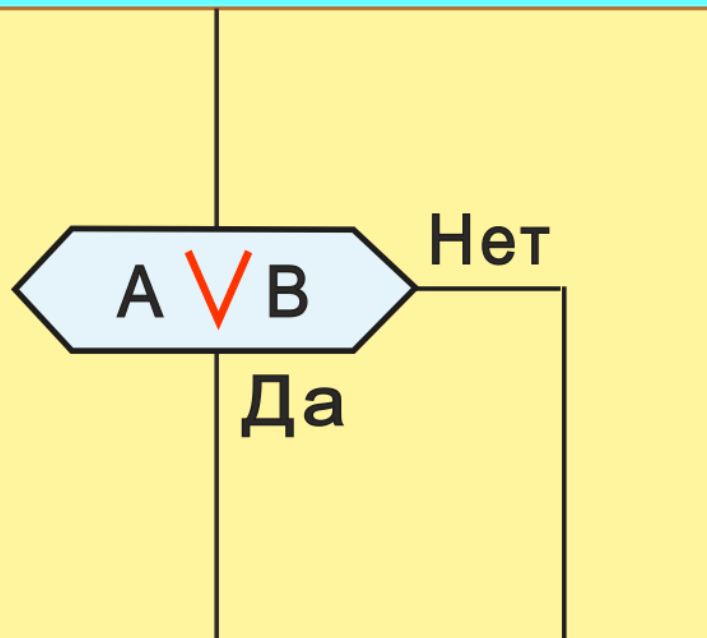
(И, ИЛИ, НЕ)

**провоцирует
появление ошибок**

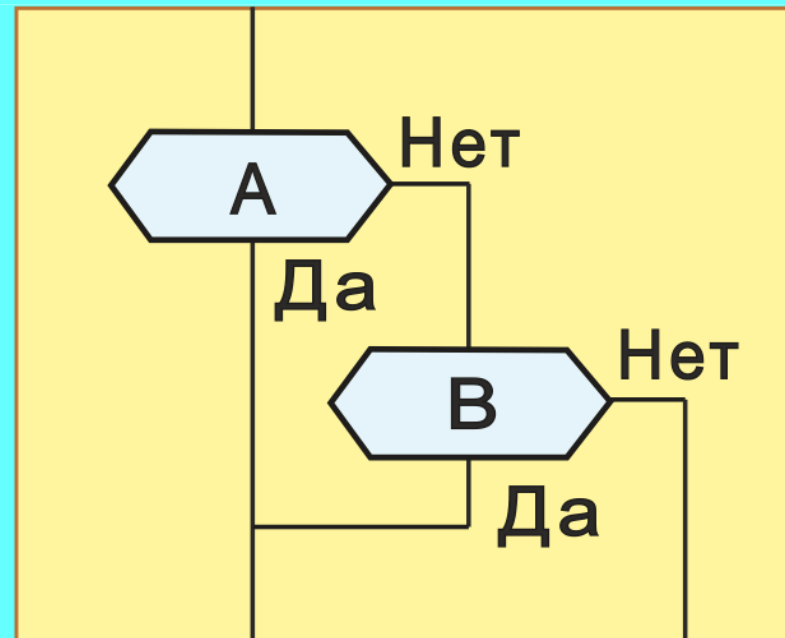
Как удалить знак конъюнкции & который провоцирует появление ошибок?



Как удалить знак ДИЗЪЮНКЦИИ **V** который провоцирует появление ошибок?



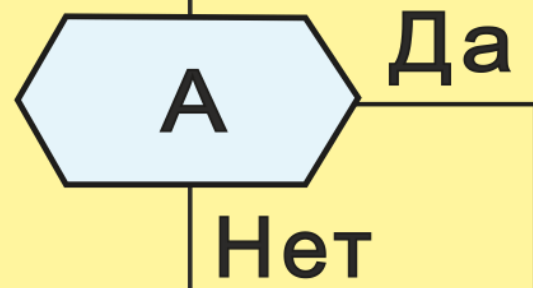
=



Как удалить знак отрицания который провоцирует появление ошибок?



=



В языке ДРАКОН логические связки

& V ¬

(И, ИЛИ, НЕ)

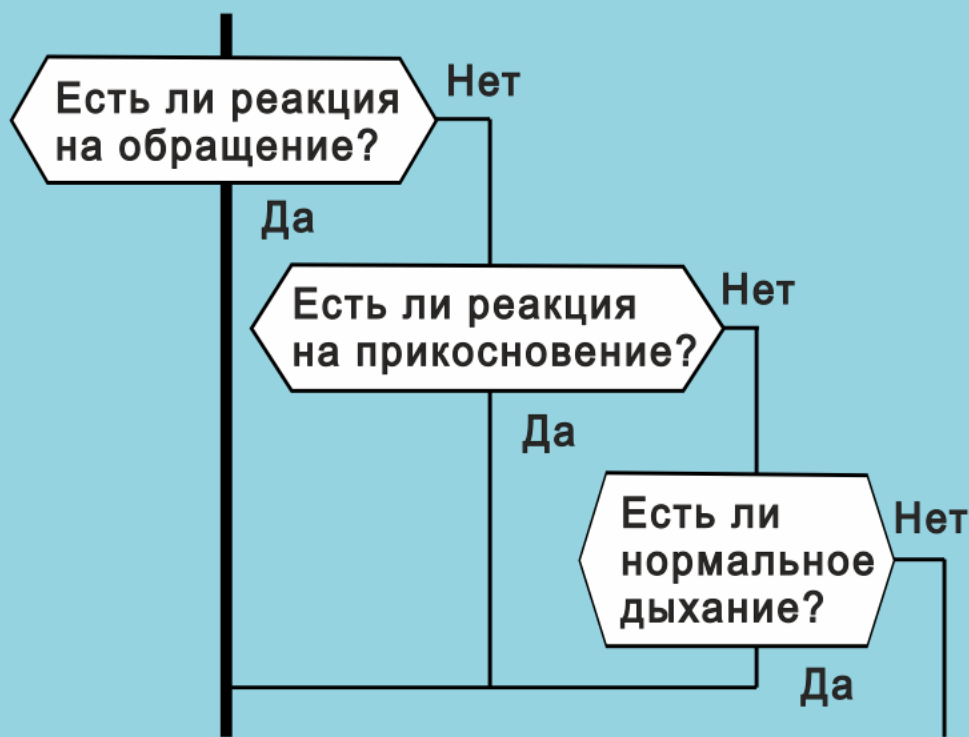
становятся ненужными.

**Вместе с ними исчезают
и вызванные ими
ошибки**

Не рекомендуется использовать пропозициональные переменные

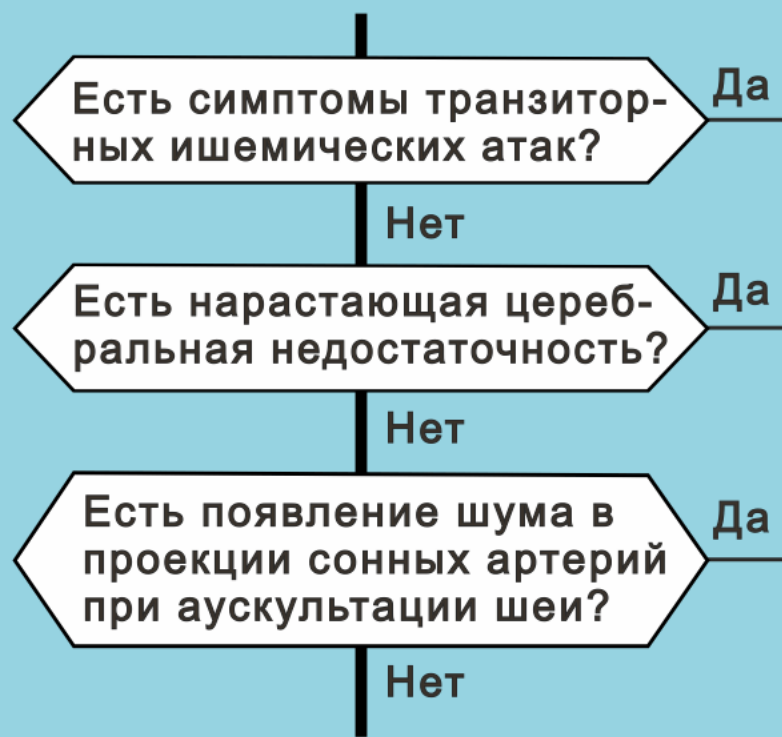
СТАНДАРТНАЯ СХЕМА «ИЛИ»

Её следует использовать
для позитивных вопросов



НЕСТАНДАРТНАЯ СХЕМА «ИЛИ»

Её следует использовать
для негативных вопросов



Неклассическая логика высказываний

Запрещено использовать:

- Пропозициональные переменные
- Пропозициональные связки
 $\&$ $\vee$ $\neg$ (И, ИЛИ, НЕ)
- Слова «Истина» и «Ложь».
Вместо них «Да» и «Нет»

Недостаток линейной Булевой алгебры

Она скрывает часть проблем

Теорема. Если один выход логического фрагмента есть результат вычисления логической функции Z , то другой выход вычисляет $\neg Z$.

**Пропозициональная формула – выражение,
построенное из пропозициональных
переменных с помощью пропозициональных
связок $\&$ $\vee$ $\neg$ (Это общеизвестно)**

**Линейная пропозициональная формула Z
не показывает свою инверсию $\neg Z$**

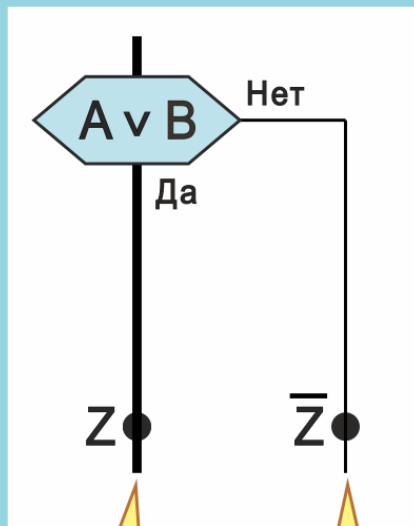
**Визуальная пропозициональная
формула Z показывает разом
оба инверсных выхода:**

1) выход Z

2) выход $\neg Z$

Три белых квадрата — три визуальных пропозициональных формулы. У каждой формулы два выхода: главный и инверсный

ФОРМУЛА
«ИЛИ»

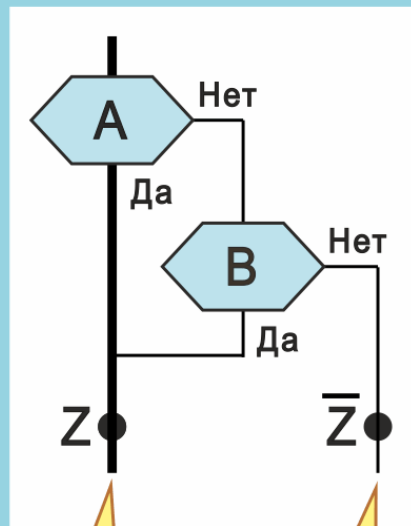


Главный
выход

Инверсный
выход

=

СТАНДАРТНАЯ
СХЕМА «ИЛИ»

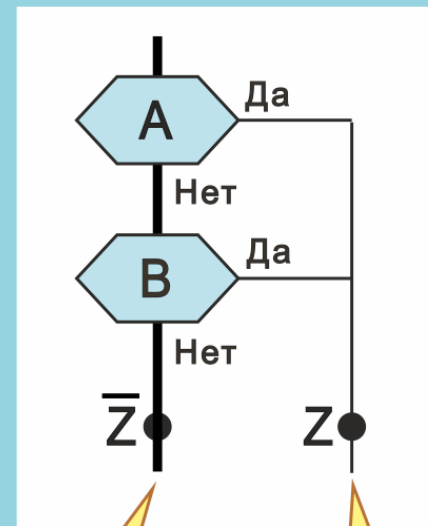


Главный
выход

Инверсный
выход

=

НЕСТАНДАРТНАЯ
СХЕМА «ИЛИ»



Инверсный
выход

Главный
выход

$$Z = A \vee B$$

$$\bar{Z} = \overline{A \vee B}$$

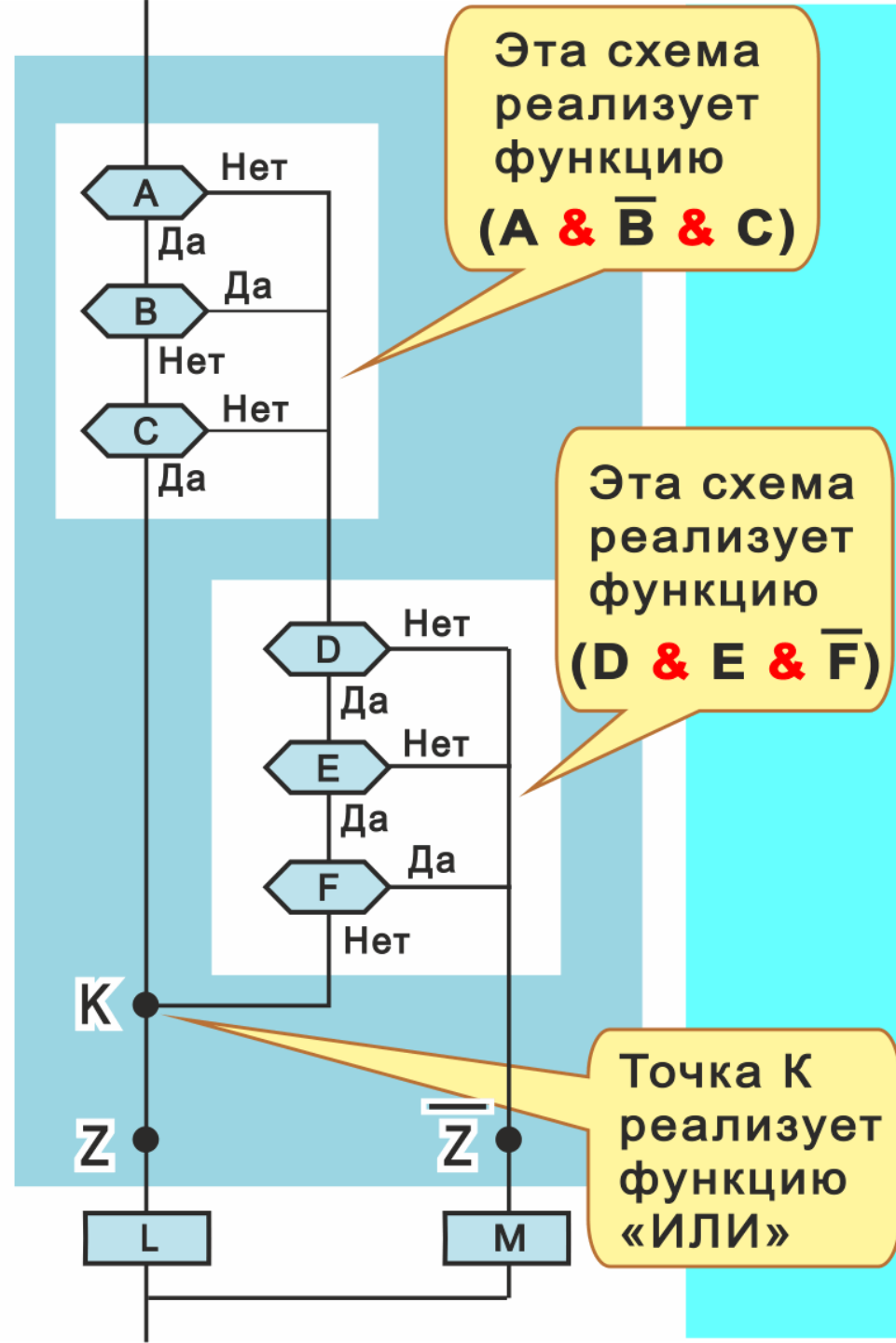
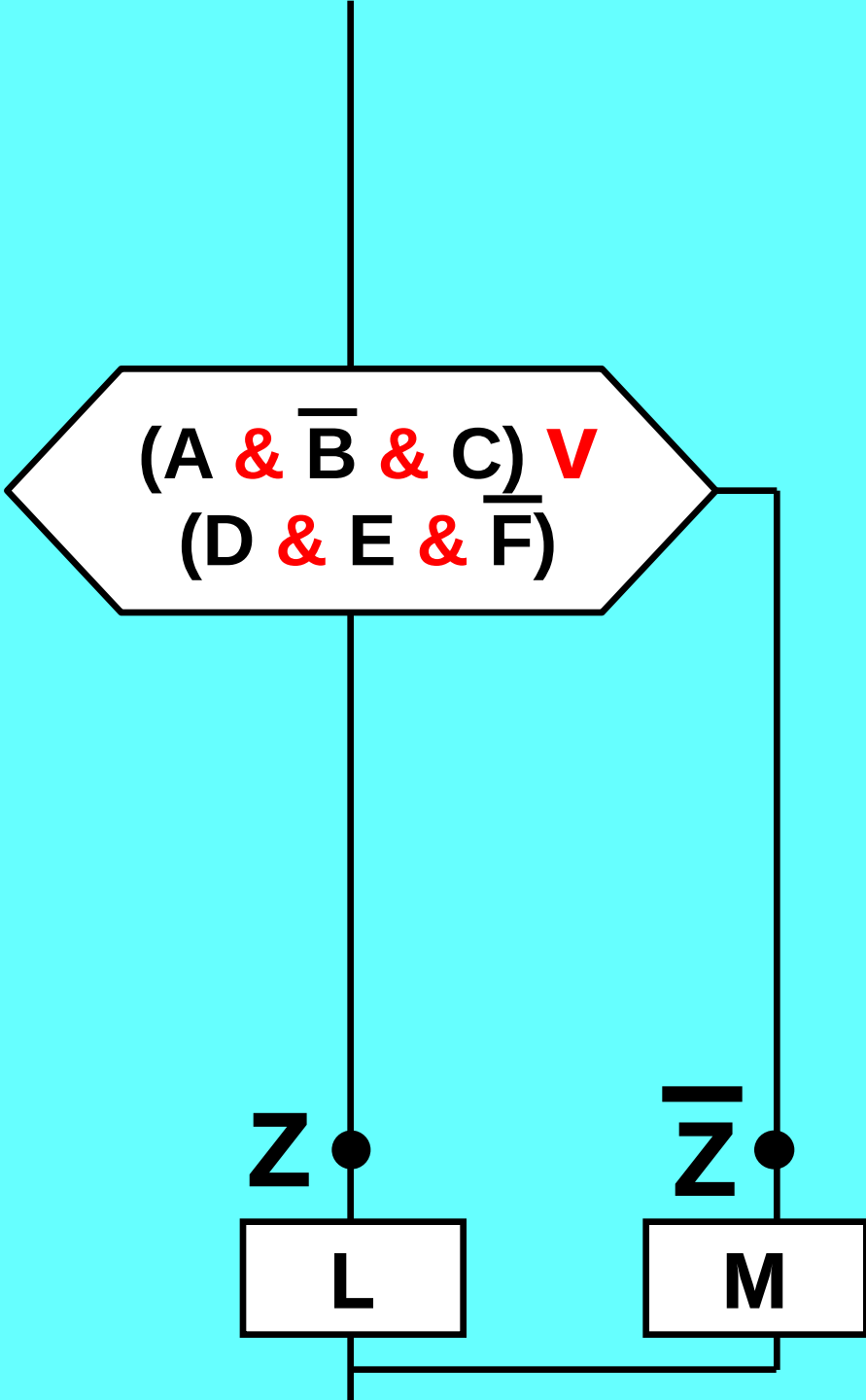
$$Z = A \vee B$$

$$\bar{Z} = \bar{A} \& \bar{B}$$

$$\bar{Z} = \bar{A} \& \bar{B}$$

$$Z = A \vee B$$

Первый закон де Моргана $\bar{A} \& \bar{B} = \overline{A \vee B}$



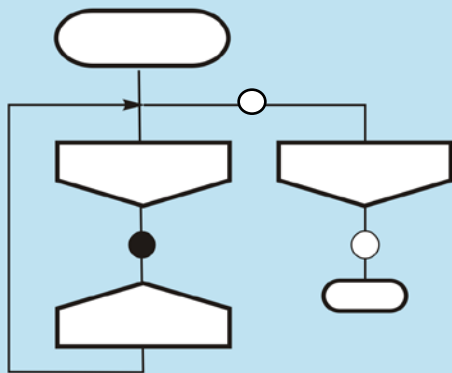
**Недостаток линейной логики
высказываний состоит в том, что:**

**Любая формула классической
логики высказываний
демонстрирует лишь одно
высказывание из инверсной
пары высказываний и
игнорирует парное
(инверсное к нему)
высказывание.**

Математическая логика

- Визуальное логическое исчисление
- Метод визуального логического вывода

Аксиома-
силуэт

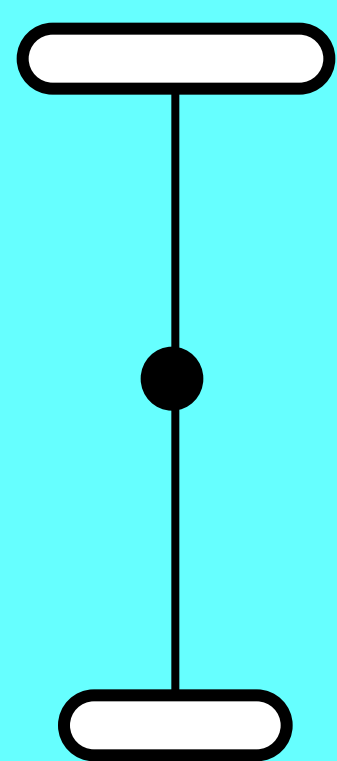


Аксиома-
примитив

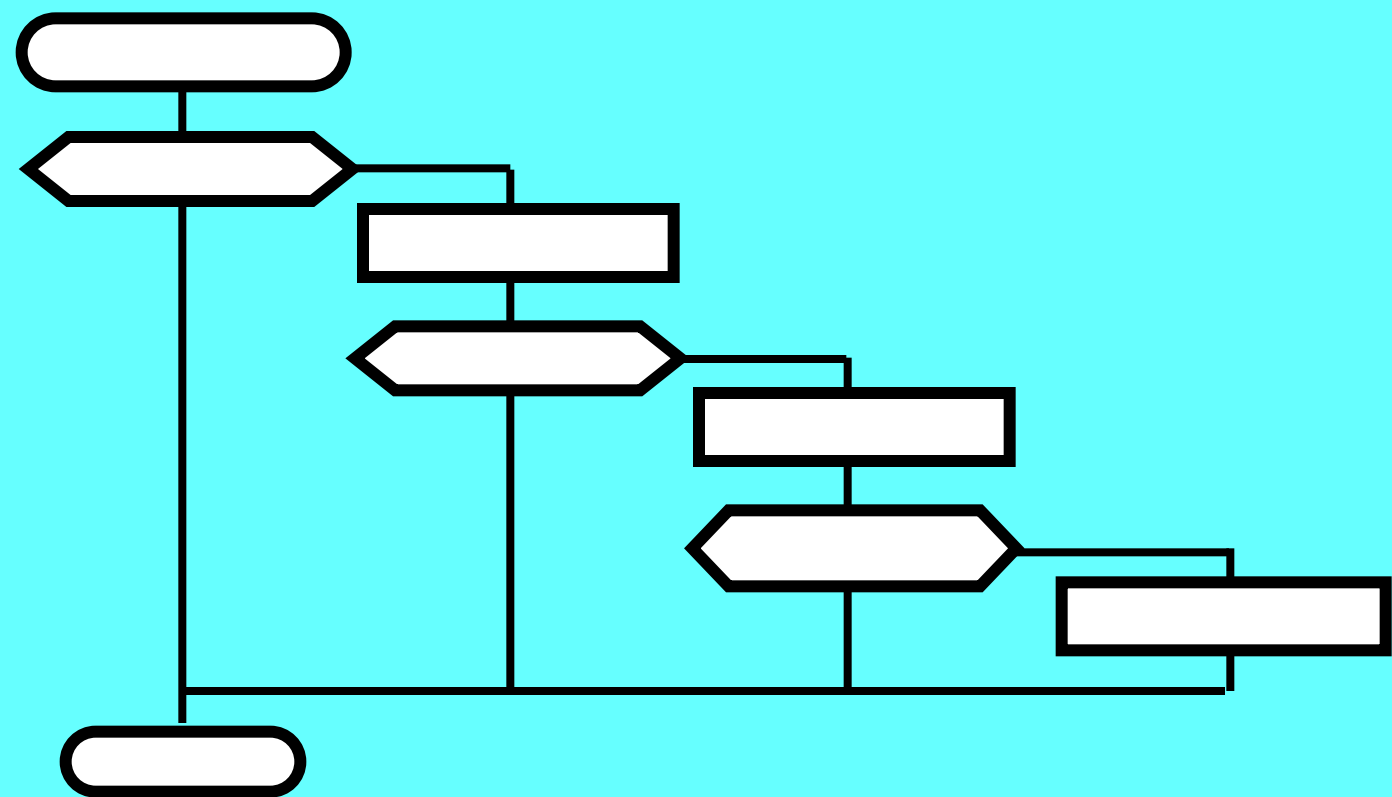


**Из этих аксиом методом
визуального логического вывода
выводятся строго доказанные
теоремы (шампур-схемы)**

Аксиома-примитив



Теорема-примитив



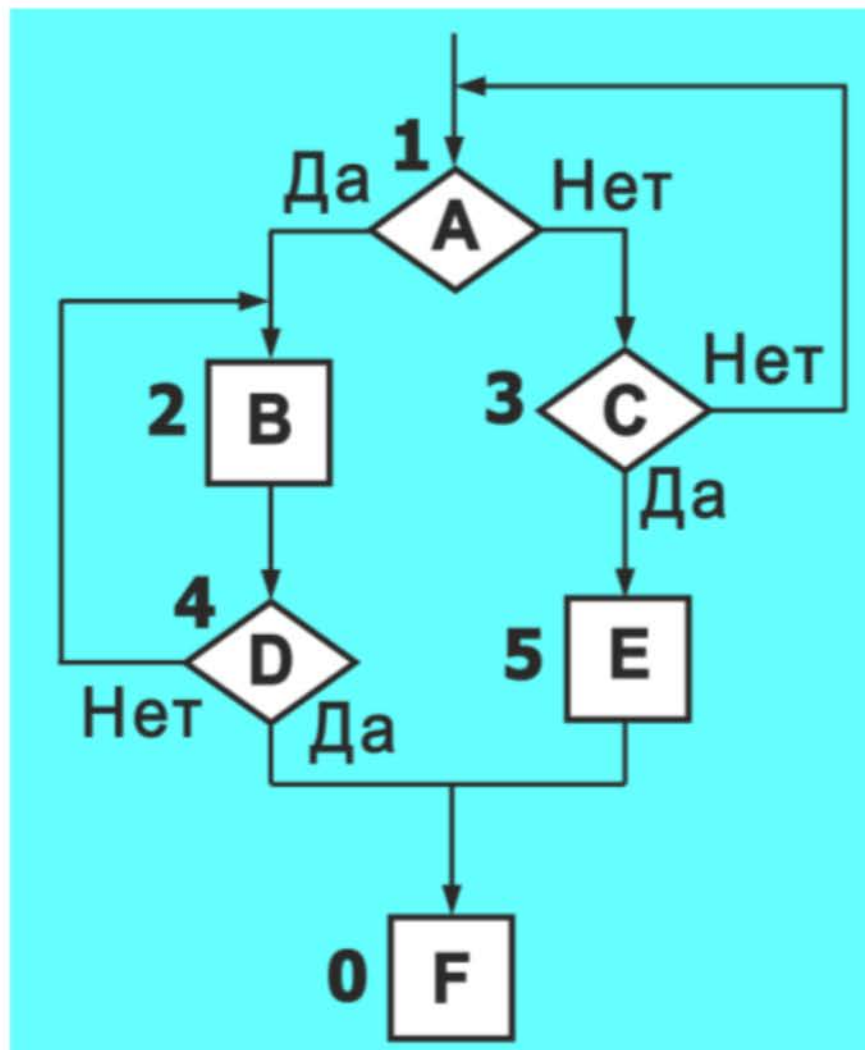
Метод Ашкрофта-Манны

**преобразования неструктурной
программы в структурную**

Метод введения переменной состояния

По Эдварду Йодану

Пронумеруем блоки неструктурной программы

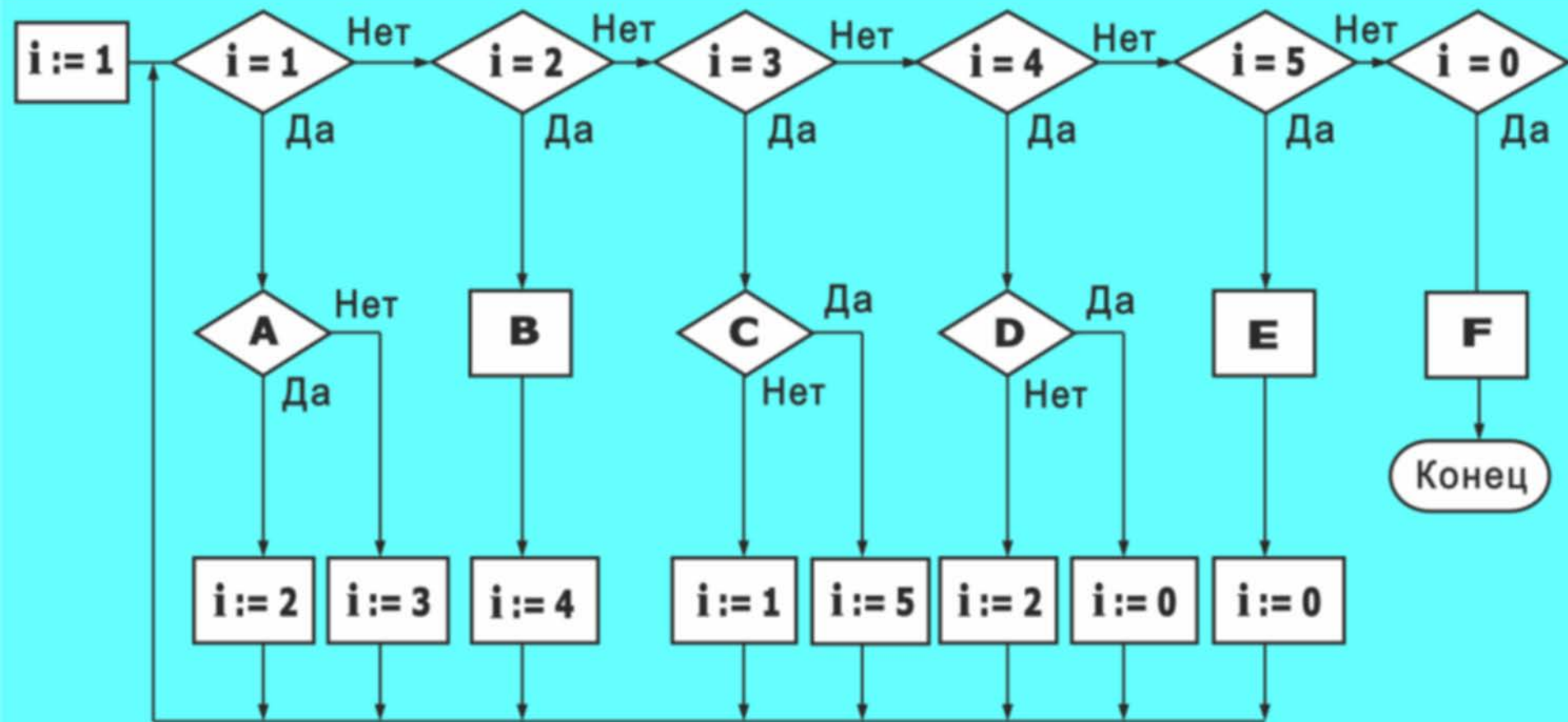


**Добавим переменную
состояния**

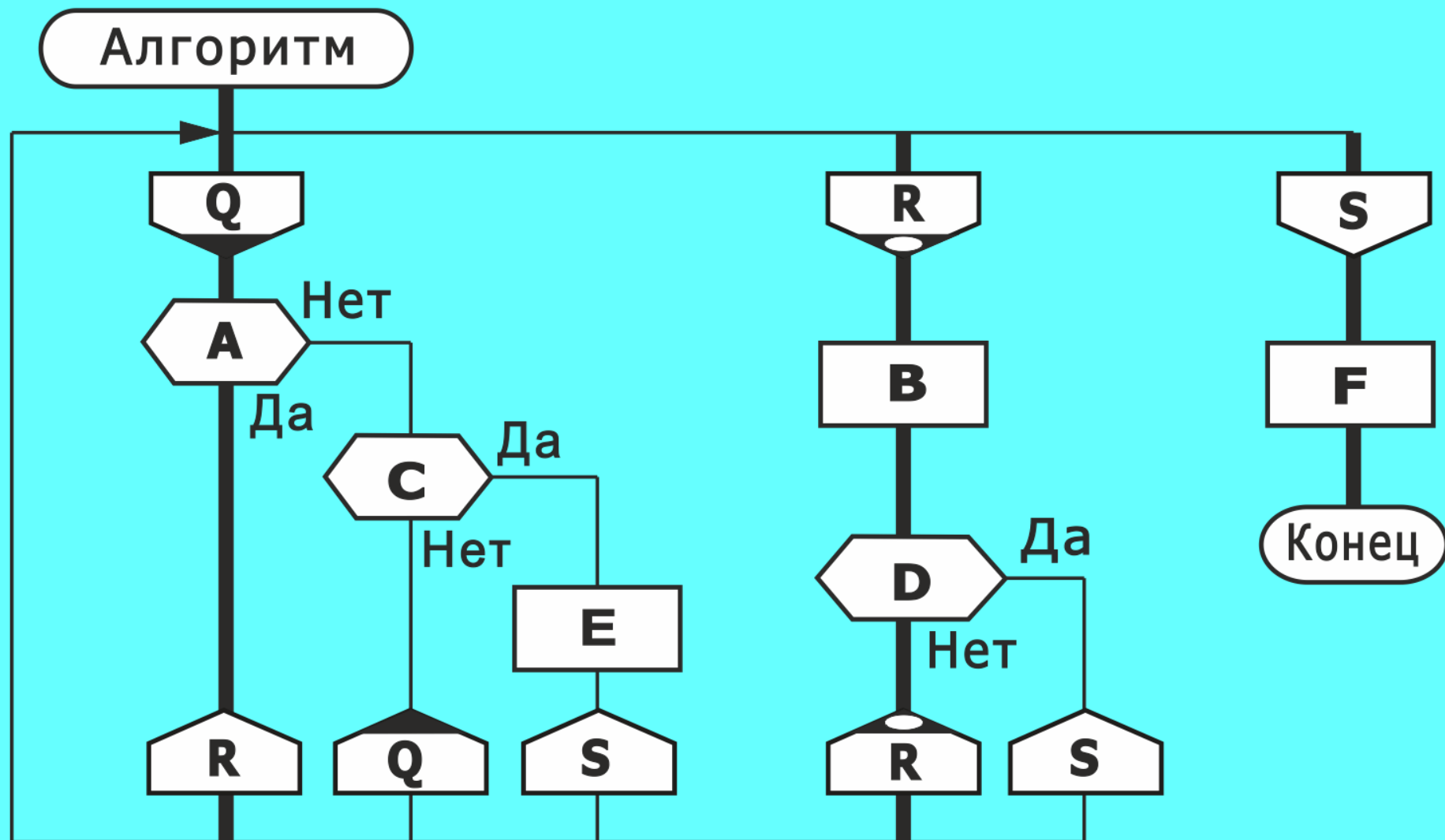
$i = 1, 2, 3, 4, 5$

и служебные блоки.

**После этого повернем
блок-схему на 90°
и зеркально отразим**

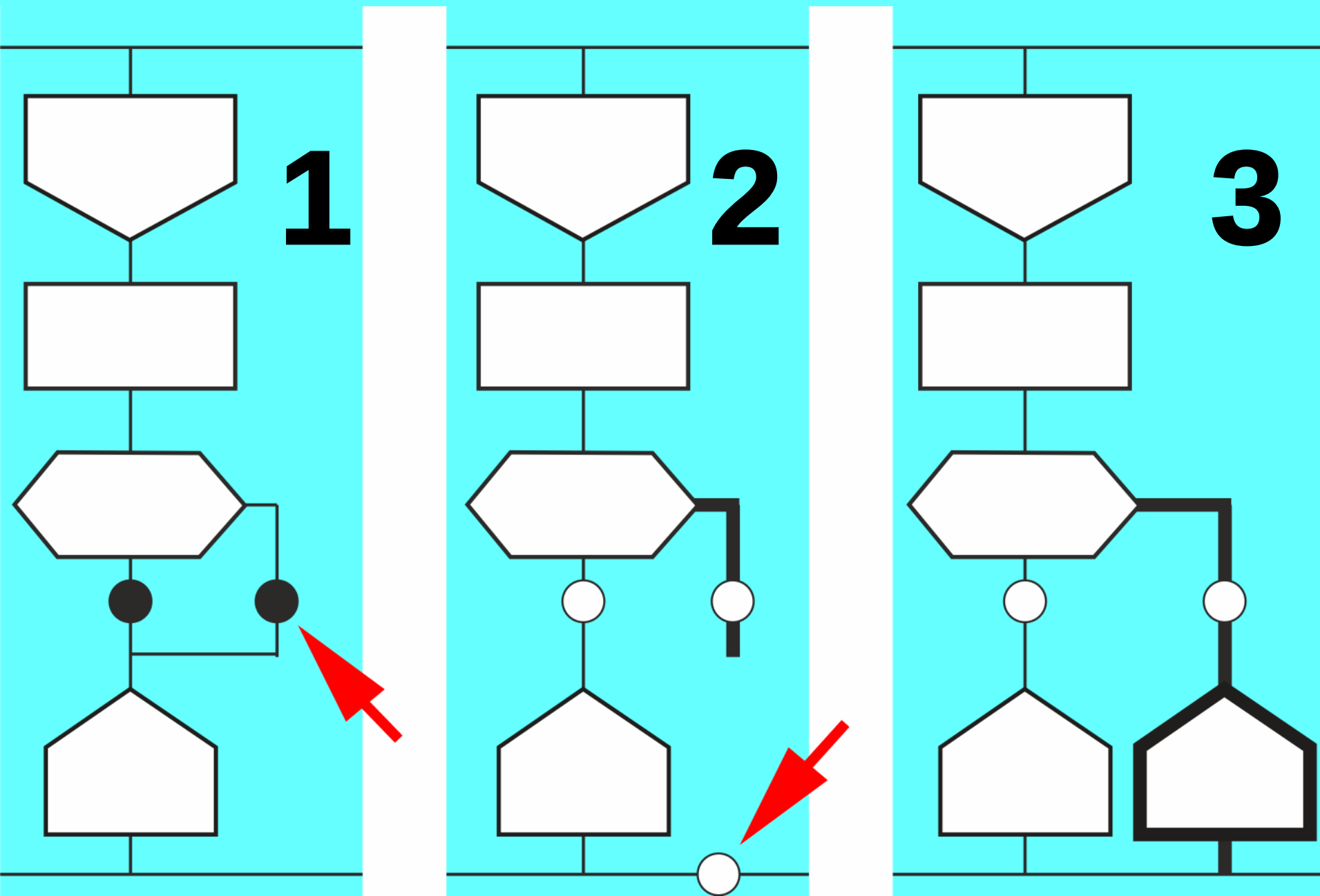


Структурный силуэт, полностью соответствующий исходной неструктурной программе



Операции с лианой

Заземление лианы



Тезис Ланды-Непейводы
«Необходимо различать
алгоритм и
алгоритмическое
предписание, имеющее
внешнюю форму
алгоритма, но
включающее не до конца
определенные шаги»

- **Алгоритмы**
- **Медицинские алгоритмы** Тавровский
- **Потоки работ (workflows)**
- **Бизнес-процессы, стартапы**
- **Алгоритмические предписания**

Владимир Паронджанов

Почему врачи убивают и калечат пациентов

или Зачем врачу блок-схемы алгоритмов?



Предисловие доктора медицинских наук,
профессора, члена-корреспондента РАН
Геннадия Порядина

Алгоритмы для эффективного клинического мышления

Пневмонэктомия

Чек-лист

Пуски ракет-носителей и разгонных блоков, при создании которых использовался язык ДРАКОН, производятся или производились с шести космодромов мира:

- ❖ Плесецк**
- ❖ Байконур**
- ❖ Kourou (Французская Гвиана)**
- ❖ Плавучий космодром
«Морской старт»**
- ❖ Naro (Южная Корея)**
- ❖ Восточный**

Семейство ДРАКОН-языков

язык Дракон + язык Си	= гибридный язык Дракон-Си
язык Дракон + язык Java	= гибридный язык Дракон-Java
язык Дракон + язык Си#	= гибридный язык Дракон-Си#
язык Дракон + язык Питон	= гибридный язык Дракон-Питон
язык Дракон + язык Tsl	= гибридный язык Дракон-Tsl
язык Дракон + язык Javascript	= гибридный язык Дракон-Javascript
язык Дракон + язык Lua	= гибридный язык Дракон-Lua
язык Дракон + язык Ада	= гибридный язык Дракон-Ада
язык Дракон + язык Erlang	= гибридный язык Дракон-Erlang
язык Дракон + язык Оберон	= гибридный язык Дракон-Оберон
язык Дракон + язык Ассемблер	= гибридный язык Дракон-Ассемблер
и т. д.	

В 1996 году Государственный комитет Российской Федерации по высшему образованию включил изучение языка ДРАКОН в программу курса «Информатика»

Примерная программа дисциплины «Информатика». Издание официальное. — М.: Госкомвуз, 1996. — 21 с.

http://drakon.su/_media/biblioteka/progr_drakon.pdf

Журавлев Юрий Иванович

- ❑ **ДРАКОН-конструктор**
Геннадия Тышова
- ❑ **Онлайн ДРАКОН-конструктор**
Степана Митькина
- ❑ **Офлайн ДРАКОН-конструктор**
Степана Митькина
- ❑ **ДРАКОН-конструктор**
Эдуарда Ильченко
- ❑ **ДРАКОН-конструктор**
Артема Бразовского
- ❑ **ДРАКОН-конструктор**
Леонида Эйсмонта

**Паронджанов
Владимир
Данилович**

vdp2007@bk.ru

Тел. 8-916-111-91-57

Сайт [http: // drakon.su](http://drakon.su)