Введение в программирование

Лекция 1

Лопатин Александр

2015

О чем курс

- урезанная версия одноименного универовского курса (обычно рассчитанного на 1—2 семестра)
- основы и упоминания из других курсов

Expectations

- получим кругозор в программировании
- узнаем основы нескольких языков
- научимся базовым техникам
- на практике применим некоторые техники

Как построен курс

- разбит так, чтобы не вызвать передоз информацией
- одно занятие в неделю на 2—3 часа (включает лекцию и практику)
- 10 недель ($\approx 2\frac{1}{3}$ месяца)
- домашки на не более 2-х часов в неделю
- один мелкий проект на 2 недели (2 викенда)
- домашки и проект будут оцениваться

Исполнитель

- повар
- военнослужащий
- гитарист
- коммерческая компания
- вычислительная система (компьютер, смартфон, бытовой прибор и т.п.)

Инструкции исполнителя

- **пункты рецепта** («42. Положить ложку сахара») для повара
- **приказы** («нале-во!») для военнослужащего
- **аккорды** для гитариста
- **инструкции процессора** («сложить два числа») для вычислительной системы

Алгоритм

Конечная последовательность **инструкций исполнителя** направленная на решение задачи

- **рецепт** для повара
- **лабораторная работа** для студента
- **текст песни** для вокалиста
- **бизнес-план** коммерческой компании
- **шаги воспроизведения проблемы** для тестировщика
- **компьютерная программа** для вычислительной системы

С точки зрения данных

Ввод $\rightarrow$ Обработка $\rightarrow$ Вывод

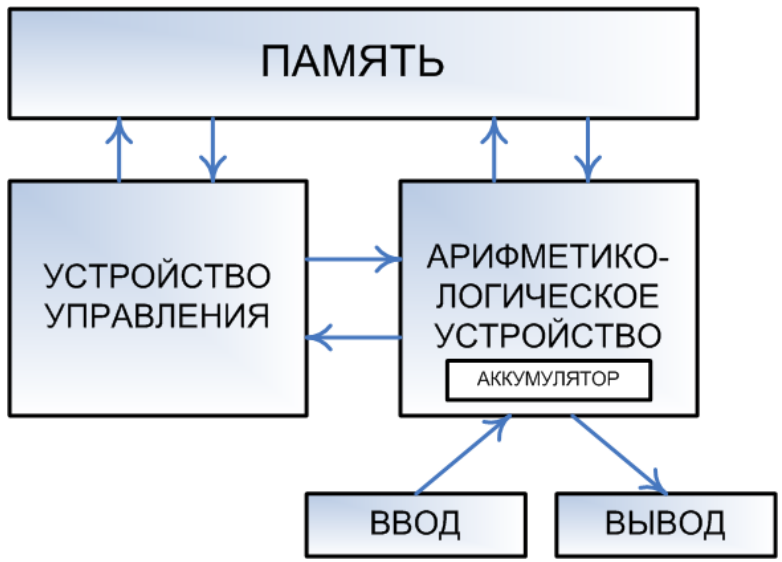
Интерфейс

Означает «взаимодействие»
Нужно для осуществления ввода/вывода данных

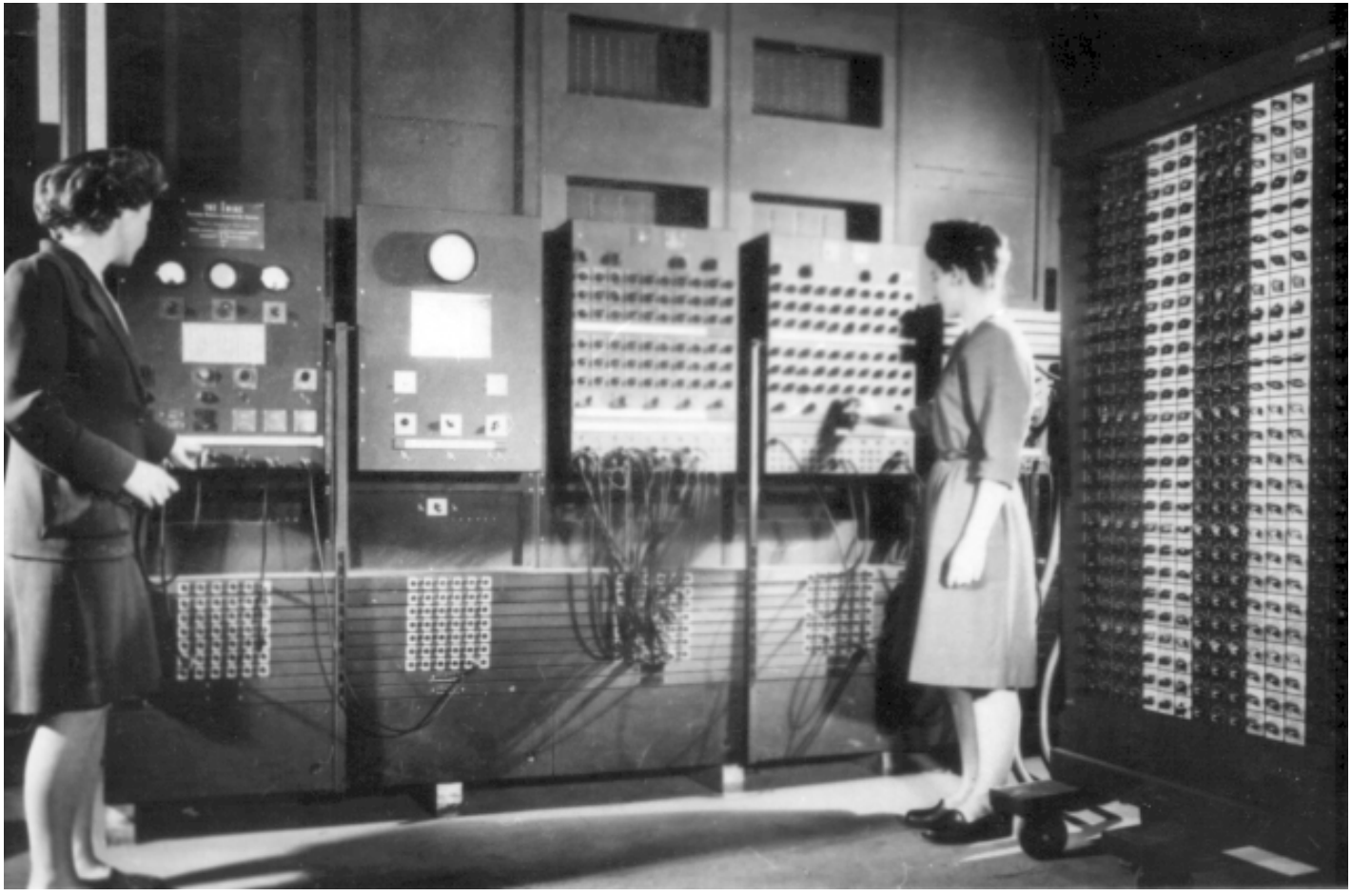
- Пользовательский (UI — User Interface)
  - Графический пользовательский интерфейс, Graphical User Interface (GUI)
  - Интерфейс коммандной строчки, Command-Line Interface (CLI)
- Программный (API — Application Interface)
  - подпрограммы, модули, библиотеки
  - сетевые протоколы (например HTTP)
  - много других

CLI может использоваться как API

1945: Архитектура фон Неймана (Von Neumann architecture)



1946: ENIAC

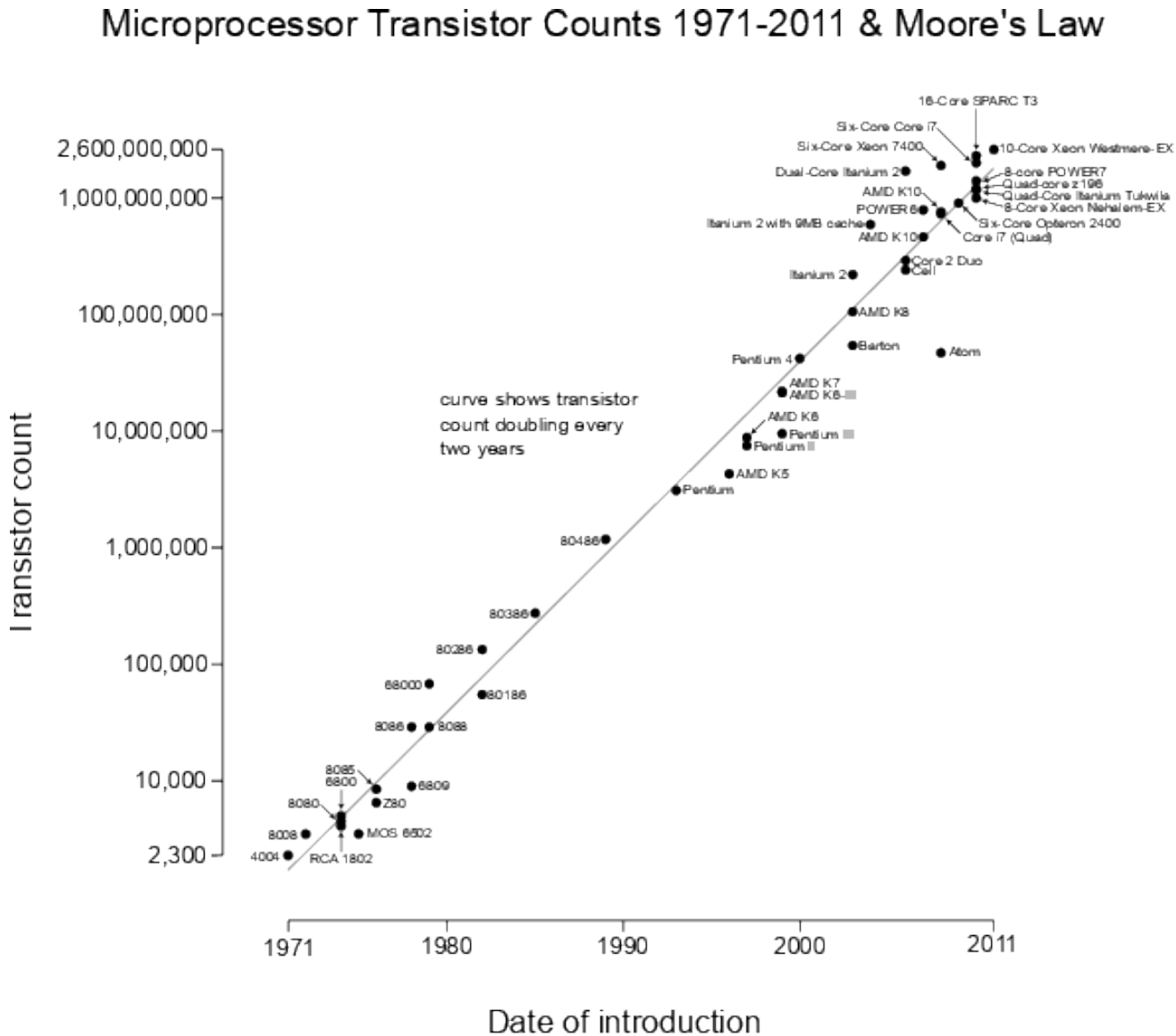


1964: IBM System/360

В 60-е значение слова «Хакер» еще не было испорчено журналистами

 Стивен Леви — Хакеры: Герои компьютерной революции

1975: Закон Мура



Закон прекратил действовать — пришло время новых идей по увеличению производительности железа

- выполнять трудоемкие операции на других устройствах (например GPU)
- объединять несколько ядер процессоров в один
- объединять несколько компьютеров в вычислительный кластер
- проектировать концептуально другие выч. системы, необязательно на основе фон Неймовской архитектуры
  - квантовые компьютеры
  - клеточные компьютеры
  - ...

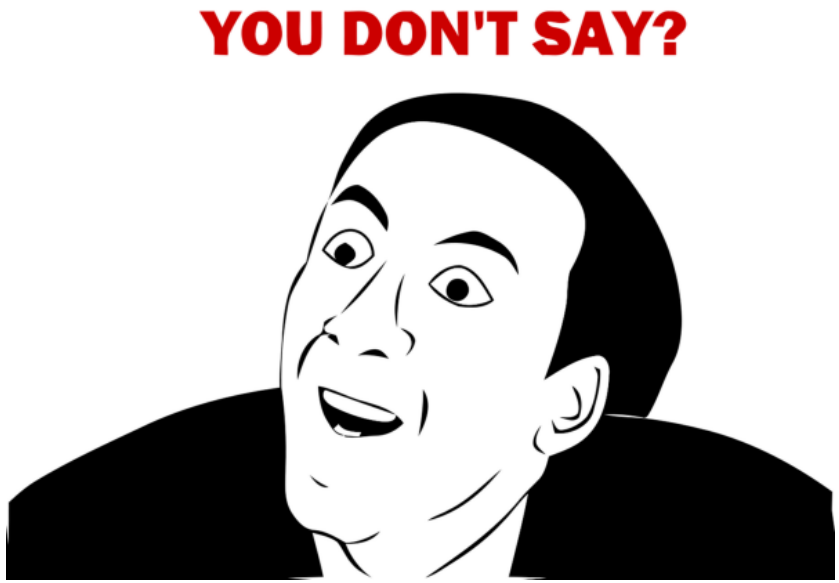
С ростом производительности железа растет и сложность задач

Сложность программ тоже растет

Появляется много специализаций в IT, по аналогии с врачами

Многоуровневые системы

Системы состоят из подсистем (из слоёв / уровней)

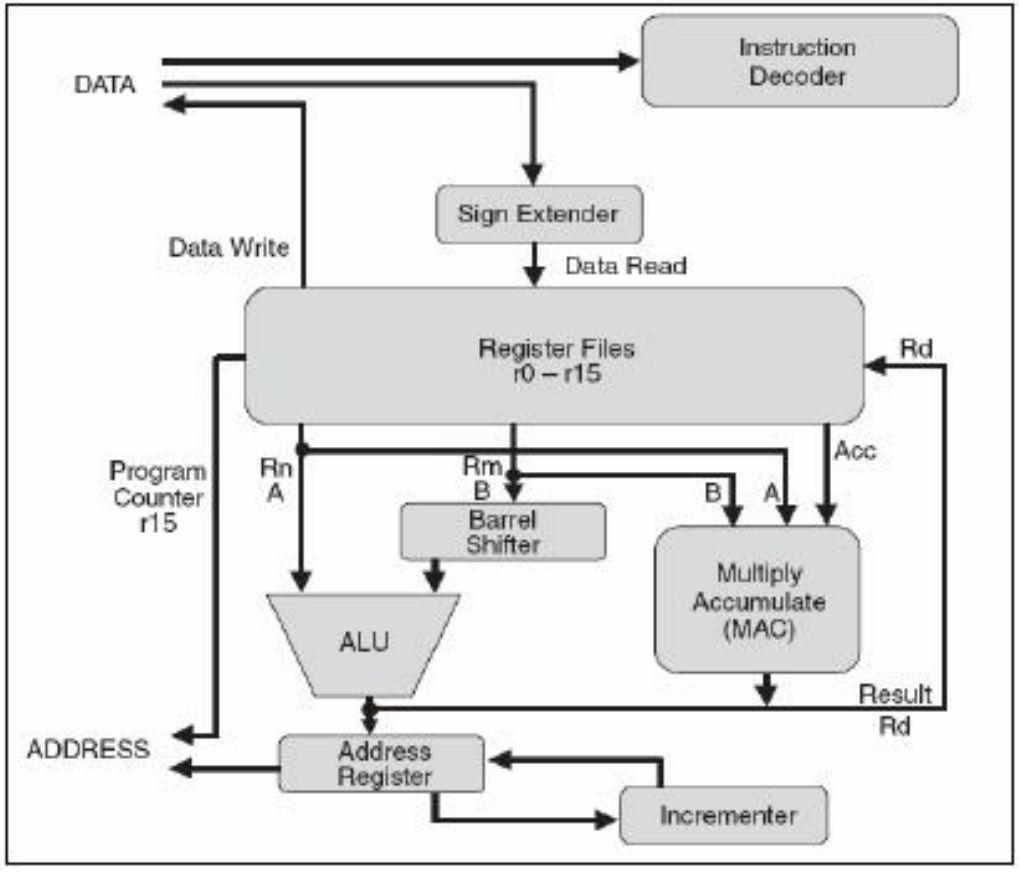


Уровни (levels / layers) в реальной жизни

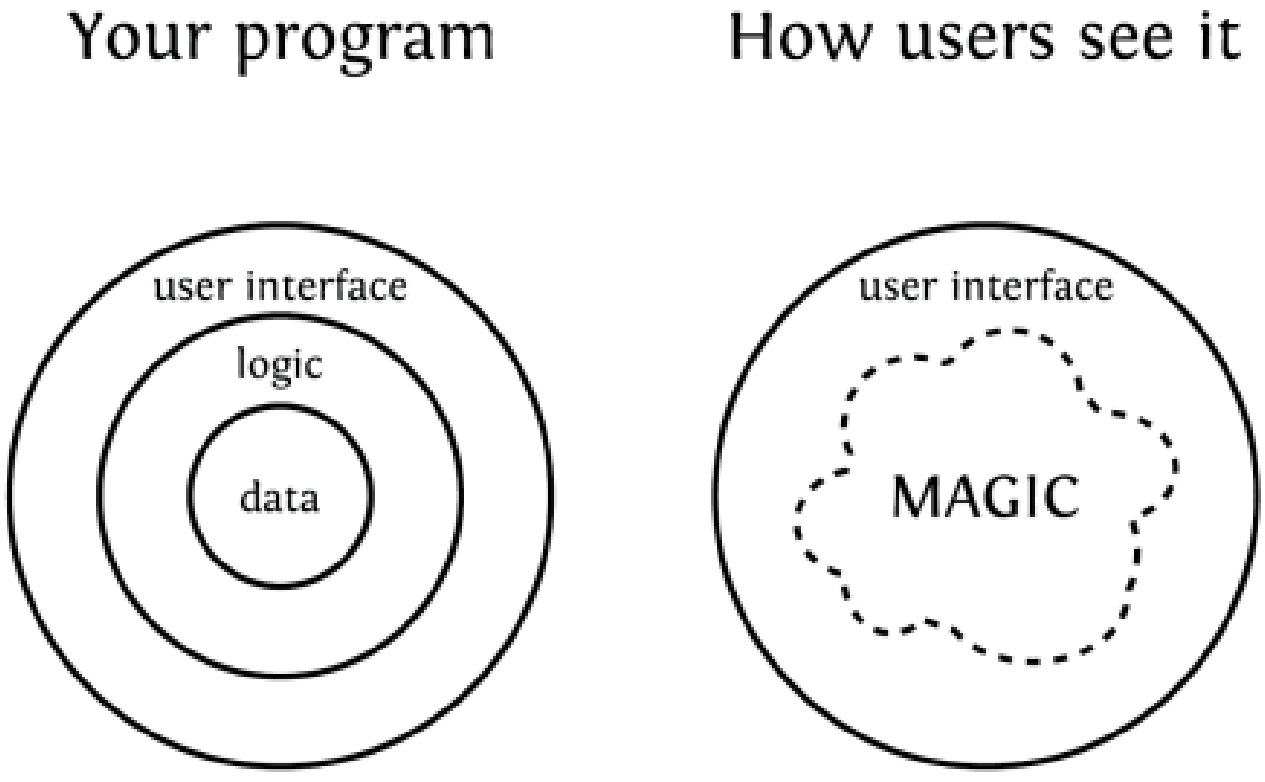
Уровни характеризуются **обязанностями** и определяют **кто над кем главнее (выше)**

- Директор конторы (высокий уровень)
  - Художник (средний уровень)
  - Программист (средний уровень)
    - Младший программист (низкий уровень)

Каждая железяка состоит из множества более низкоуровневых железок



Софт тоже многоуровневый



Декларативное

Императивное

Императивное программирование (Imperative Programming)

Написание **алгоритмов** путём перечисления **инструкций исполнителя**

Декларативное программирование (Declarative Programming)

Описание желаемого результата, вместо алгоритма получения этого результата.

Например запрос к базе данных:

```
Выбрать имена абитуриентов
поступающих на специальность «Программная Инженерия»
с сортировкой по сумме баллов по убыванию
```

Подробнее рассмотрим ближе к концу курса

Декларативное

Императивное

Алгоритмическое

Блок-схемы, словесное описание

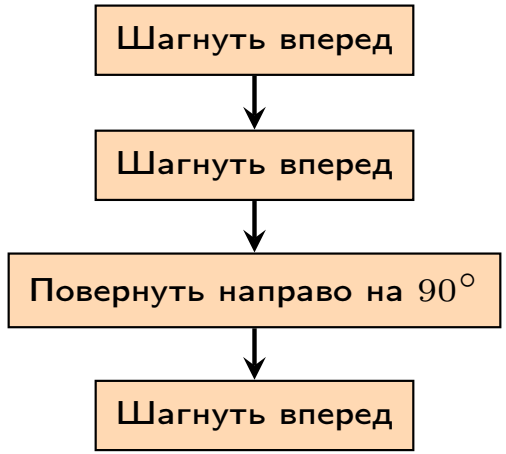
Линейный алгоритм

1. Шагнуть вперед
2. Шагнуть вперед
3. Повернуть направо на 90°
4. Шагнуть вперед

Алгоритм с ветвлением

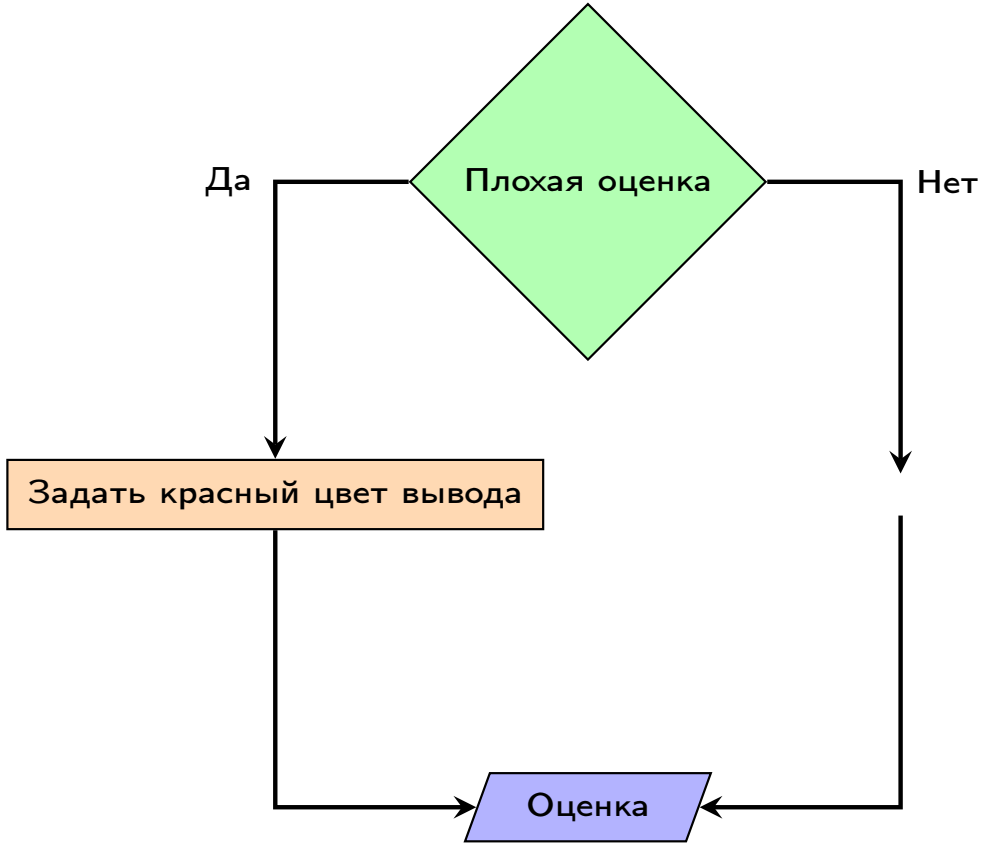
1. Если небо светлое — перейти к п. 2 иначе к п. 3
2. Напечатать «Сейчас день»
3. Перейти к п.5
4. Напечатать «Сейчас ночь»
5. Напечатать «Ваш Капитан Очевидность»

Линейный алгоритм



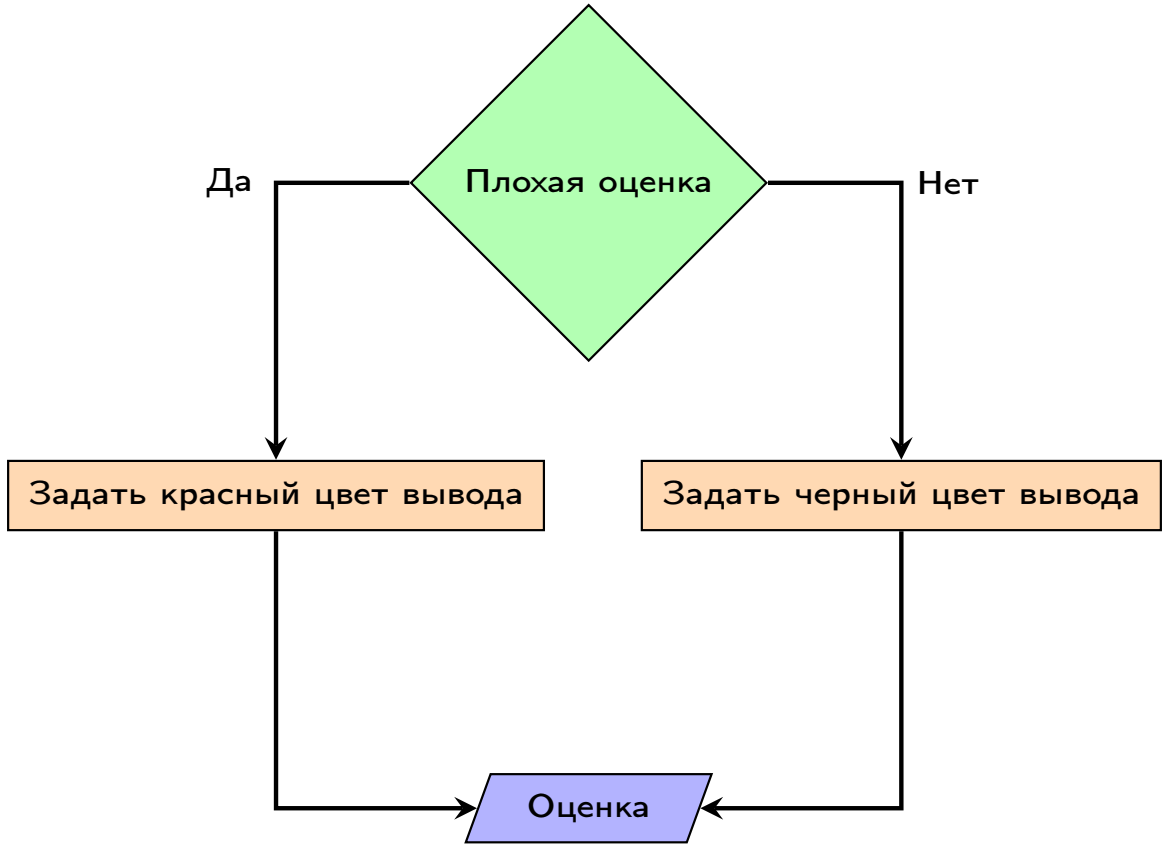
Алгоритм с ветвлением

Бывает с одной веткой

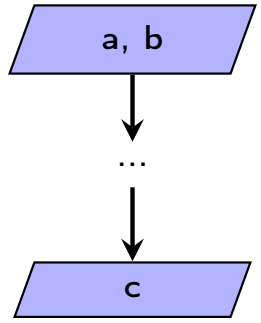


Алгоритм с ветвлением

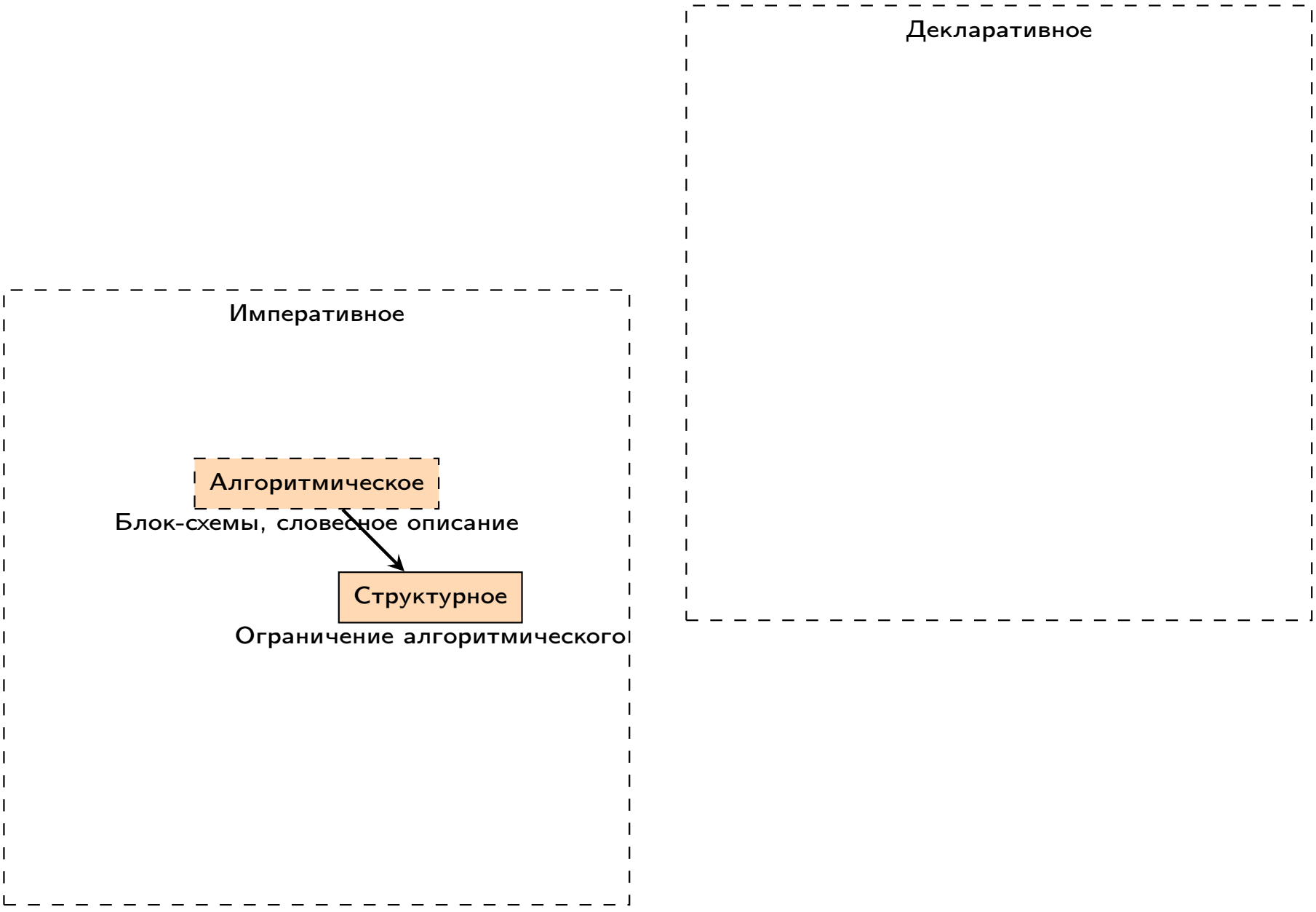
И с двумя ветками



Практика

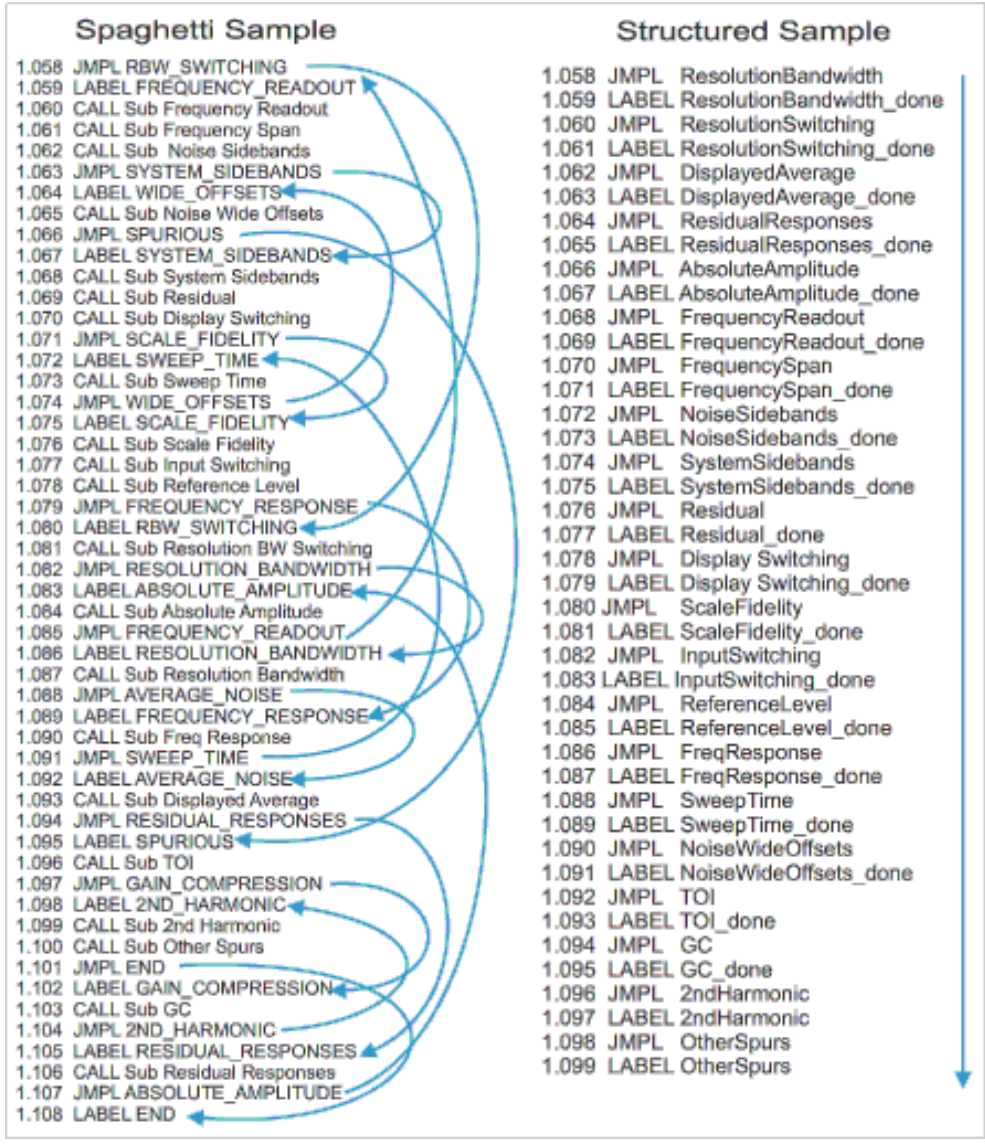


1. Нарисовать блок-схему к программе, которая берет в качестве ввода числа a и b и решает уравнение $a = b \cdot c$. Вывести решение (значение c). (Подсказка: сначала нужно вручную преобразовать уравнение)
2. Предусмотреть обработку ошибки деления на ноль к предыдущему заданию. Выводить «Ошибка: деление на ноль» вместо решения в этом случае
3. Написать словесное описание получившегося алгоритма

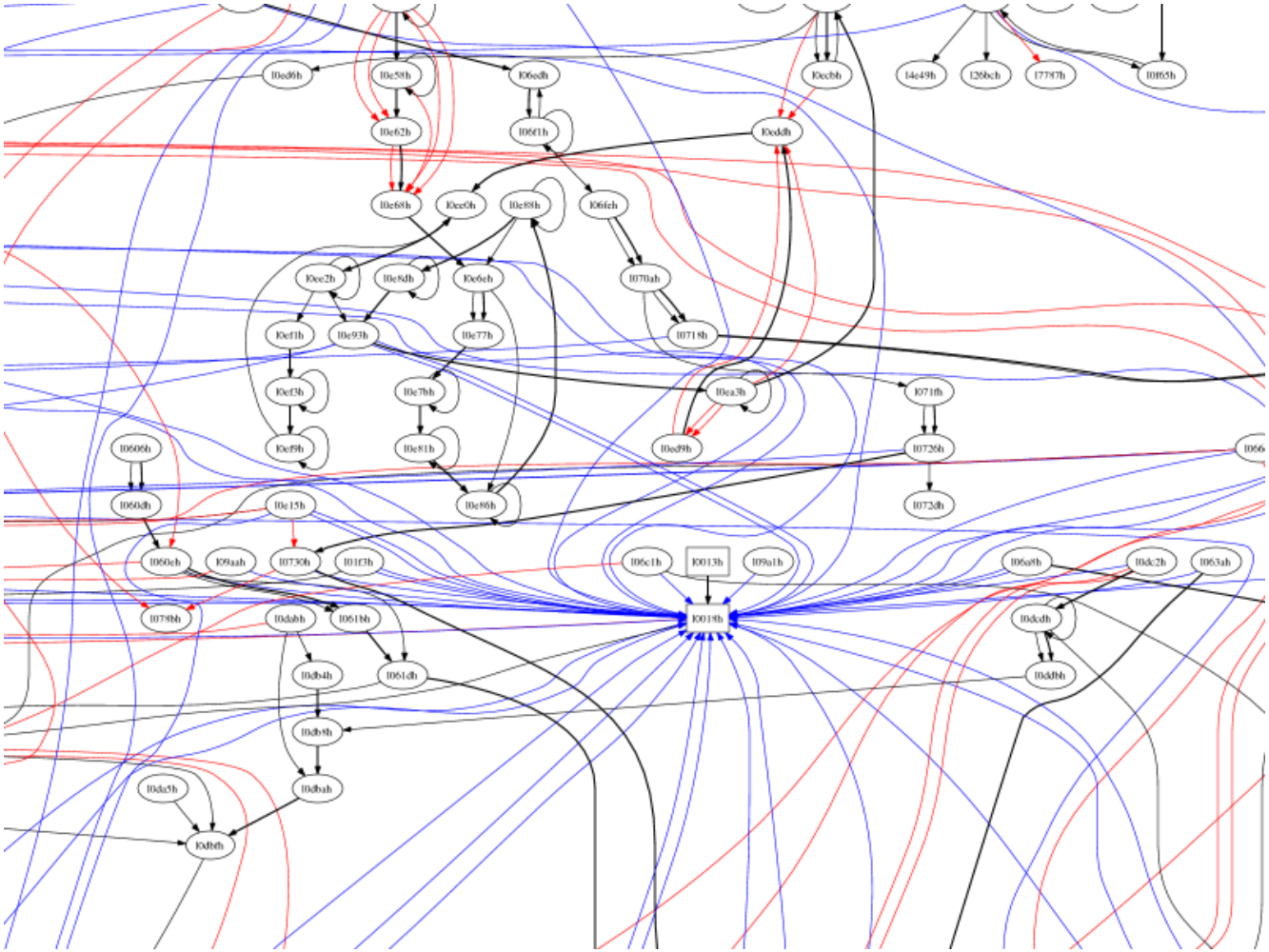

Декларативное
Императивное
Алгоритмическое
Блок-схемы, словесное описание
Структурное
Ограничение алгоритмического

Структурное программирование (Structured Programming)

- отказ от меток и безусловного перехода
- использование двух **управляющих структур**
  - условие
  - цикл


Spaghetti Sample
1.058 JMPL RBW_SWITCHING
1.059 LABEL FREQUENCY_READOUT
1.060 CALL Sub Frequency Readout
1.061 CALL Sub Frequency Span
1.062 CALL Sub Noise Sidebands
1.063 JMPL SYSTEM_SIDEBANDS
1.064 LABEL WIDE_OFFSETS
1.065 CALL Sub Noise Wide Offsets
1.066 JMPL SPURIOUS
1.067 LABEL SYSTEM_SIDEBANDS
1.068 CALL Sub System Sidebands
1.069 CALL Sub Residual
1.070 CALL Sub Display Switching
1.071 JMPL SCALE_FIDELITY
1.072 LABEL SWEEP_TIME
1.073 CALL Sub Sweep Time
1.074 JMPL WIDE_OFFSETS
1.075 LABEL SCALE_FIDELITY
1.076 CALL Sub Scale Fidelity
1.077 CALL Sub Input Switching
1.078 CALL Sub Reference Level
1.079 JMPL FREQUENCY_RESPONSE
1.080 LABEL RBW_SWITCHING
1.081 CALL Sub Resolution BW Switching
1.082 JMPL RESOLUTION_BANDWIDTH
1.083 LABEL ABSOLUTE_AMPLITUDE
1.084 CALL Sub Absolute Amplitude
1.085 JMPL FREQUENCY_READOUT
1.086 LABEL RESOLUTION_BANDWIDTH
1.087 CALL Sub Resolution Bandwidth
1.088 JMPL AVERAGE_NOISE
1.089 LABEL FREQUENCY_RESPONSE
1.090 CALL Sub Freq Response
1.091 JMPL SWEEP_TIME
1.092 LABEL AVERAGE_NOISE
1.093 CALL Sub Displayed Average
1.094 JMPL RESIDUAL_RESPONSES
1.095 LABEL SPURIOUS
1.096 CALL Sub TOI
1.097 JMPL GAIN_COMPRESSION
1.098 LABEL 2ND_HARMONIC
1.099 CALL Sub 2nd Harmonic
1.100 CALL Sub Other Spurs
1.101 JMPL END
1.102 LABEL GAIN_COMPRESSION
1.103 CALL Sub GC
1.104 JMPL 2ND_HARMONIC
1.105 LABEL RESIDUAL_RESPONSES
1.106 CALL Sub Residual Responses
1.107 JMPL ABSOLUTE_AMPLITUDE
1.108 LABEL END
Structured Sample
1.058 JMPL ResolutionBandwidth
1.059 LABEL ResolutionBandwidth_done
1.060 JMPL ResolutionSwitching
1.061 LABEL ResolutionSwitching_done
1.062 JMPL DisplayedAverage
1.063 LABEL DisplayedAverage_done
1.064 JMPL ResidualResponses
1.065 LABEL ResidualResponses_done
1.066 JMPL AbsoluteAmplitude
1.067 LABEL AbsoluteAmplitude_done
1.068 JMPL FrequencyReadout
1.069 LABEL FrequencyReadout_done
1.070 JMPL FrequencySpan
1.071 LABEL FrequencySpan_done
1.072 JMPL NoiseSidebands
1.073 LABEL NoiseSidebands_done
1.074 JMPL SystemSidebands
1.075 LABEL SystemSidebands_done
1.076 JMPL Residual
1.077 LABEL Residual_done
1.078 JMPL Display Switching
1.079 LABEL Display Switching_done
1.080 JMPL ScaleFidelity
1.081 LABEL ScaleFidelity_done
1.082 JMPL InputSwitching
1.083 LABEL InputSwitching_done
1.084 JMPL ReferenceLevel
1.085 LABEL ReferenceLevel_done
1.086 JMPL FreqResponse
1.087 LABEL FreqResponse_done
1.088 JMPL SweepTime
1.089 LABEL SweepTime_done
1.090 JMPL NoiseWideOffsets
1.091 LABEL NoiseWideOffsets_done
1.092 JMPL TOI
1.093 LABEL TOI_done
1.094 JMPL GC
1.095 LABEL GC_done
1.096 JMPL 2ndHarmonic
1.097 LABEL 2ndHarmonic
1.098 JMPL OtherSpurs
1.099 LABEL OtherSpurs

Spaghetti is write-only code



Повторения всё же нужны

Но не такие безобразные

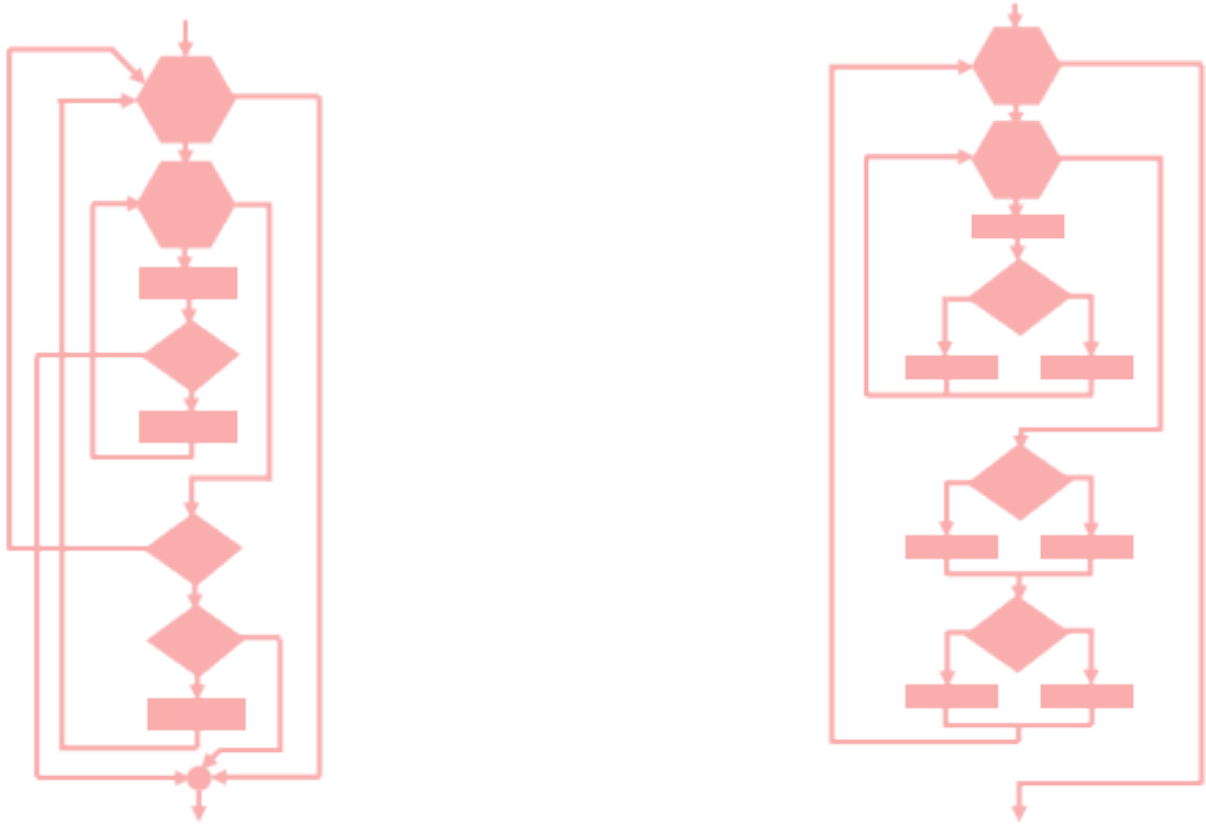
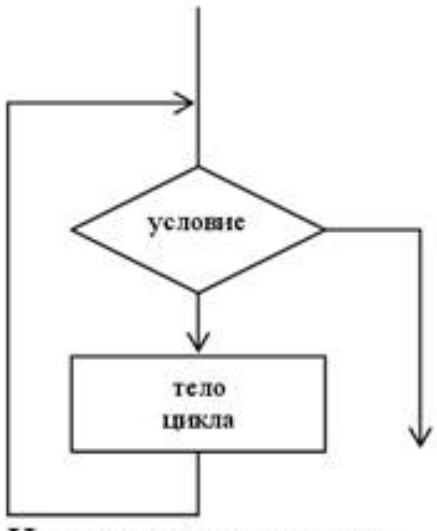
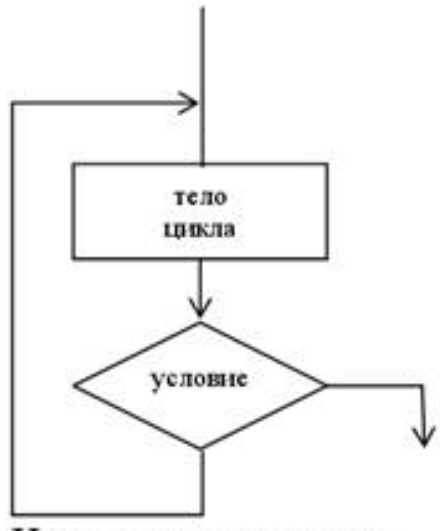


Рис.: Спагетти-код и структурный код

Циклы



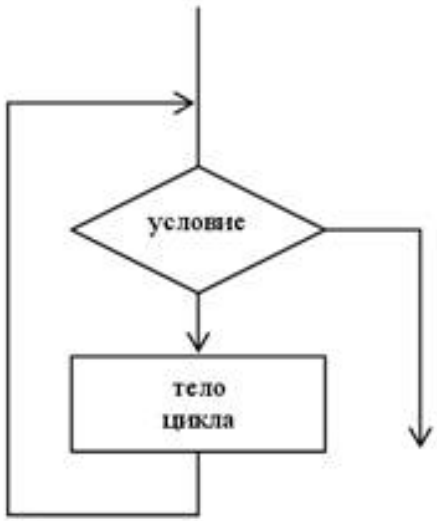
Цикл с предусловием



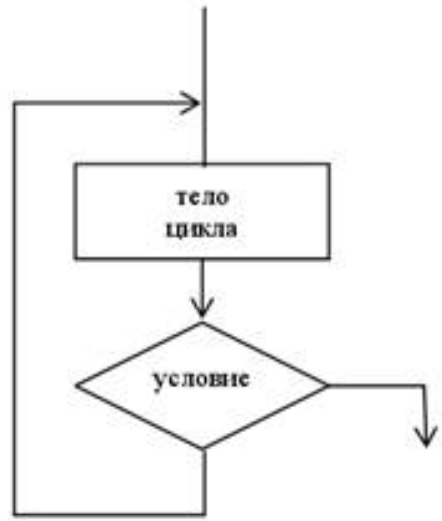
Цикл с постусловием

Цикл с предусловием (while)

1. Пока гвоздь не забит
   1.1 Поднять молоток
   1.2 Ударить молотком по гвоздю
   1.3 Перейти к п. 1



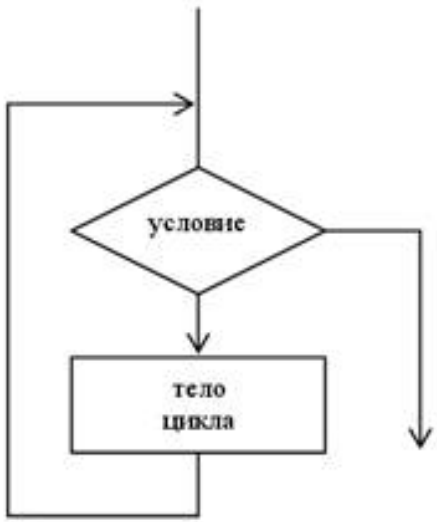
Цикл с предусловием



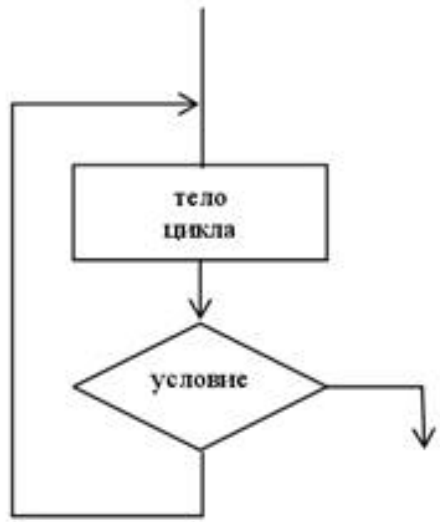
Цикл с постусловием

Цикл с постусловием (do ... while)

1. (Делать)
   1.1 Поднять молоток
   1.2 Ударить молотком по гвоздю
2. Пока гвоздь не забит — перейти к п. 1



Цикл с предусловием



Цикл с постусловием

Цикл со счетчиком (for) — частный случай цикла с предусловием

1. Инициализировать счетчик i значением 1
2. Пока i <= n
   - 2.1 Выполнить действие
   - 2.2 Выполнить другое действие
   - 2.3 ...
   - 2.4 Увеличить счетчик i на единицу
   - 2.5 Перейти к п. 2.

Всё тот же цикл for

1. Для i от 1 до n с шагом 1
   1.1 Выполнить действие
   1.2 Выполнить другое действие
   1.3 ...
   1.4 Перейти к п. 1.

Циклы могут быть вложены

1. Для i от 1 до количество_студентов с шагом 1
2. Напечатать имя i-го студента
   2.1 Для j от 1 до количество_оценок_студента с шагом 1
      2.1.1 Показать j-ю оценку i-го студента
      2.1.2 Перейти к п. 2.1
   2.2 Перейти к п. 1
3. Напечатать «такие дела»

Практика

1. Нарисовать блок-схему к алгоритму забивания гвоздя со слайда с циклом с предусловием
2. Тоже самое для слайда с циклом с постусловием
3. Тоже самое для алгоритма с вложенными циклами (подсказка: можно начать с вложенного цикла)

Домашка

Нарисовать блок-схему и написать словесное описание к алгоритму решения квадратного уравнения $ax^2 + bx + c = 0$. Входные данные: a, b и c. Выходные данные:

- либо «Корней нет»
- либо x_1
- либо x_1 и x_2

Подсказки:

- дискриминант вычисляется по формуле $D = b^2 - 4ac$
- корни уравнения вычисляются по формуле $x_{1,2} = \frac{-b \pm \sqrt{D}}{2a}$
- количество корней зависит от значений дискриминанта (3 ситуации: $D < 0$, $D = 0$ и $D > 0$)