Технологии и языки программирования

Введение

Юдинцев В. В.

Кафедра теоретической механики

22 сентября 2018 г.

САМАРСКИЙ УНИВЕРСИТЕТ
SAMARA UNIVERSITY

Содержание

Введение

Задачи курса

- Знакомство с основными структурами данных и типовыми методами обработки этих структур
- Изучение методов структурного и объектно-ориентированного программирования
- Закрепление навыков алгоритмизации и программирования на основе изучения языка программирования Python
- Создание практической базы для изучения других учебных дисциплин (“Методы вычислений”, “Компьютерный практикум по механике”, ...)

Содержание курса

I семестр (зачёт)

- Развитие языков программирования, классификация
- Обзор основных технологий программирования
- Основы программирования на языке Python

II семестр (экзамен)

- Технологии программирования на языке Python (процедурное, функциональное, ООП)
- Использование библиотек Python NumPy, SciPy для проведения научных вычислений
- Визуализация результатов: использование библиотеки matplotlib

Что такое программирование?

Программирование

Часть прикладной математики и вычислительной техники, разрабатывающая методы составления программ

Программирование

Научная и практическая деятельность по созданию программ[1]

Рейтинговые (конкурсные) списки. Бакалавриат

Бакалавриат/специалитет | Магистратура | Аспирантура

Институт/факультет	Тип	Форма обучения	Заявлений/Мест
01.03.02 Прикладная математика и информатика			
Институт информатики, математики и электроники	В рамках квоты лиц, имеющих особые права	Очная	1/1
01.03.02 Прикладная математика и информатика			
Институт информатики, математики и электроники	Общий конкурс	Очная	475/91
01.03.02 Прикладная математика и информатика			
Институт информатики, математики и электроники	По договору	Очная	11/25
01.03.03 Механика и математическое моделирование			
Институт ракетно-космической техники	Общий конкурс	Очная	132/20
01.03.03 Механика и математическое моделирование			
Институт ракетно-космической	По договору	Очная	6/25

Форма обучения

- (•) Все формы
- ( ) Очная
- ( ) Очно-заочная
- ( ) Заочная

Типы приема

- (•) Все типы
- ( ) Общий конкурс
- ( ) По договору
- ( ) Без ВИ в рамках КЦП
- ( ) В рамках квоты лиц, имеющих особые права
- ( ) Целевой прием

```
import requests
import pandas as pd
import numpy as np

address= 'http://ssau.ru/ratings/bakalavr'
html= requests.get(address,verify=False).content

df_list = pd.read_html(html,encoding='utf8', header=0)[0]

ind = df[df['Тип']=='Общий конкурс'].index

zm_str = df.iloc[ind]['Заявлений/Мест'].tolist()

zm = np.array( [ x.split('/') for x in zm_str ], dtype=float)

rd = pd.DataFrame({ 'Направление':
        df.iloc[ind-1]['Институт/факультет'].tolist(),
        'Конкурс': np.round(zm[:,0]/zm[:,1],2) })
```

```
1 newdata[["Направление","Конкурс"]].sort_values('Конкурс')[0:5]
```

	Направление	Конкурс
55	11.03.03 Конструирование и технология электрон...	1.40
57	15.03.01 Машиностроение	1.72
60	44.03.02 Психолого-педагогическое образование	2.29
56	12.03.04 Биотехнические системы и технологии	2.40
62	46.03.02 Документоведение и архивоведение	2.41

```
1 newdata[["Направление","Конкурс"]].sort_values('Конкурс',
      ascending=False)[0:5]
```

	Направление	Конкурс
40	38.03.02 Менеджмент	91.75
39	38.03.01 Экономика	55.00
41	38.03.03 Управление персоналом	53.75
42	38.03.04 Государственное и муниципальное управ...	53.75
43	38.03.05 Бизнес-информатика	37.75

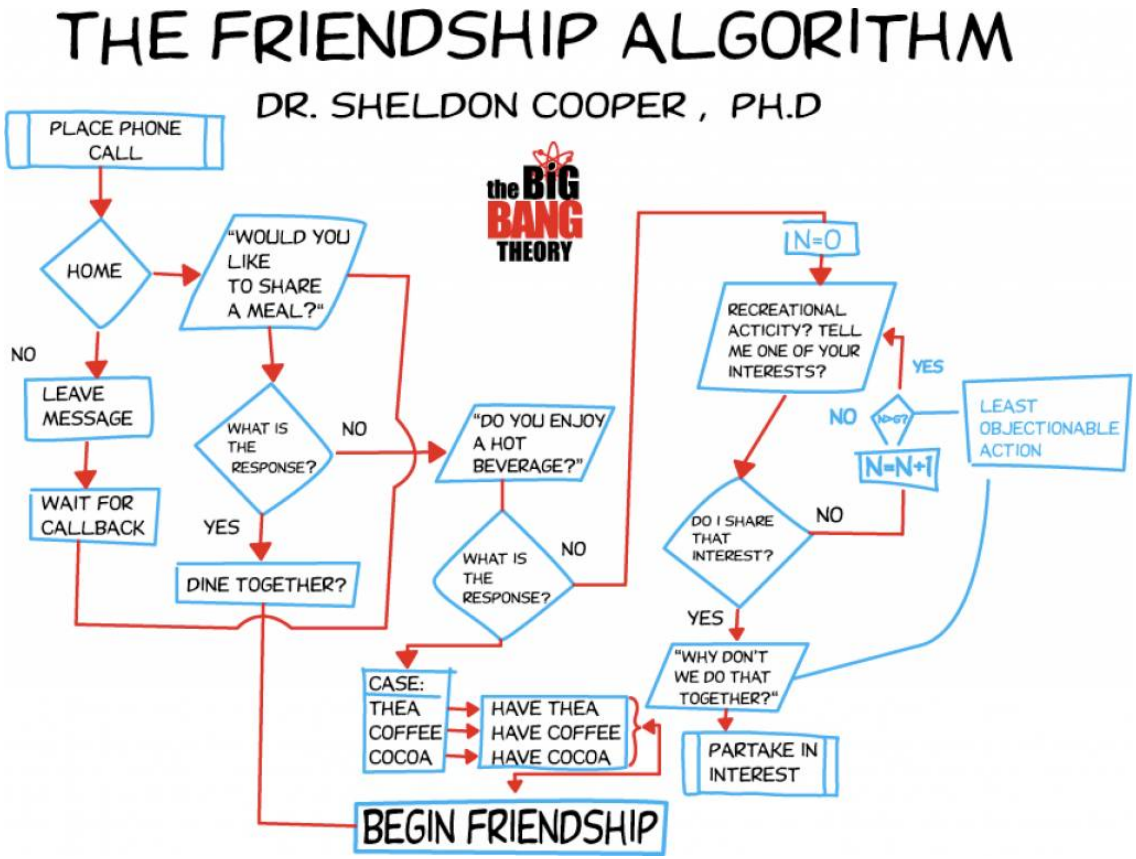
Программирование

Процесс преобразования сформулированной задачи в набор инструкций, включающий:

- анализ и разработку алгоритма решения задачи,
- оценку корректности и эффективности алгоритма,
- непосредственную реализацию алгоритма на языке программирования, который выбирается в зависимости от задачи, типа ЭВМ — написание программы.

Алгоритм

Алгоритм



Алгоритм — это набор инструкций, описывающих порядок действий исполнителя, для достижения некоторого результата

Свойства алгоритма

- Определенность и результативность
  За конечное число шагов должен быть получен результат, или доказано его отсутствие
- Массовость
  Возможность получения результата при различных исходных данных для некоторого класса сходных задач
- Формальность
  Отвлечение от содержания поставленной задачи и строгое выполнение некоторого правила, инструкции
- Дискретность
  Возможность разбиения алгоритма на отдельные элементарные действия

Формы представления алгоритма

- Словесная на неформальном языке
- Графическая (блок-схема)
- На **языках программирования**: алгоритм преобразуется в **программу**

Блок-схема: решение квадратного уравнения

Найти действительные решения квадратного уравнения ($a \neq 0$):

$$ax^2 + bx + c = 0$$

Решение

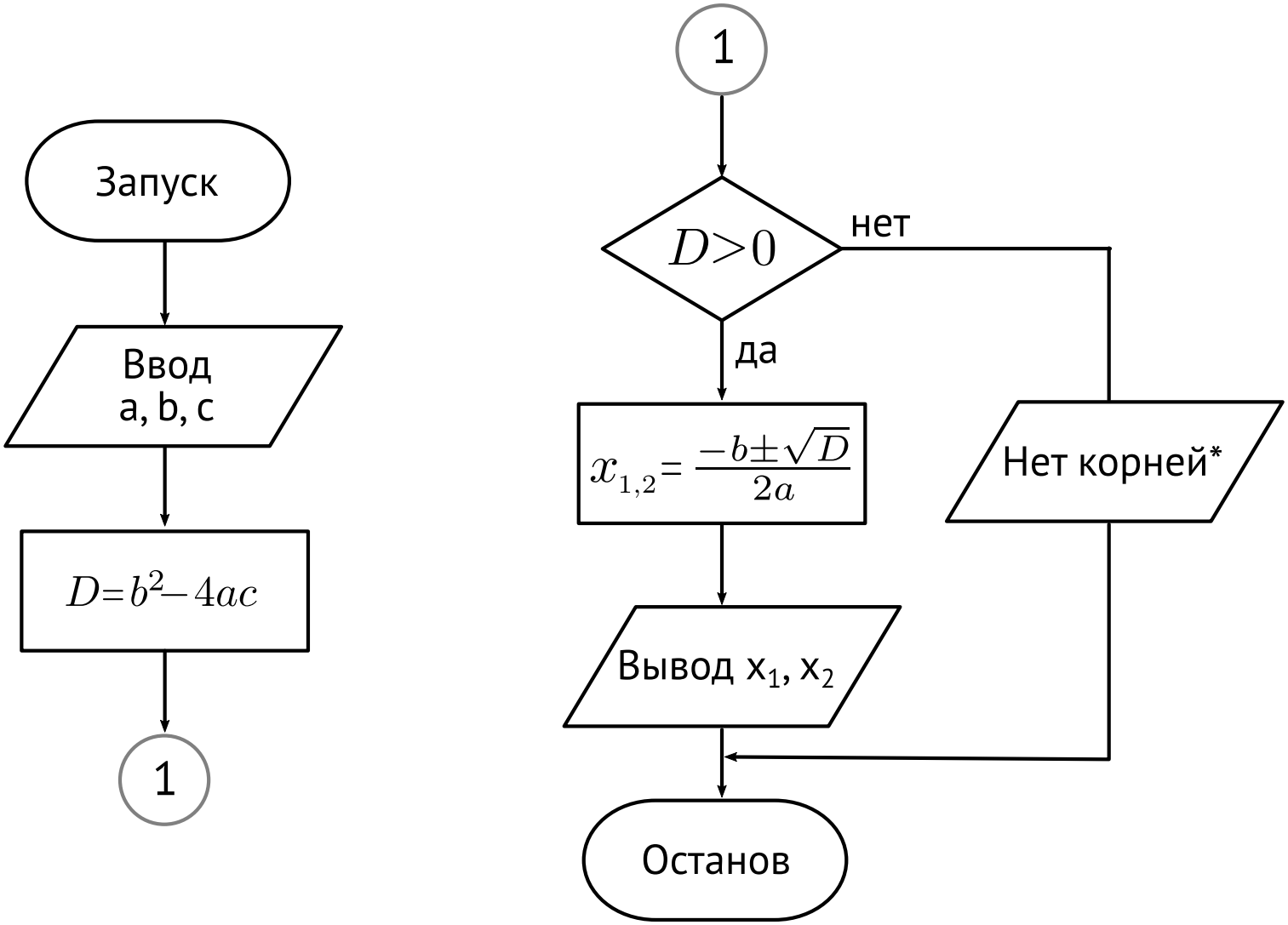
1. Определить дискриминант

$$D = b^2 - 4ac$$

2. **Если** дискриминант меньше нуля, **то** действительных корней нет
3. **Если** дискриминант больше или равен нулю, **то** есть два разных корня ($D > 0$) или два одинаковых ($D = 0$)

$$x_1 = \frac{-b + \sqrt{D}}{2a}, \quad x_2 = \frac{-b - \sqrt{D}}{2a}$$

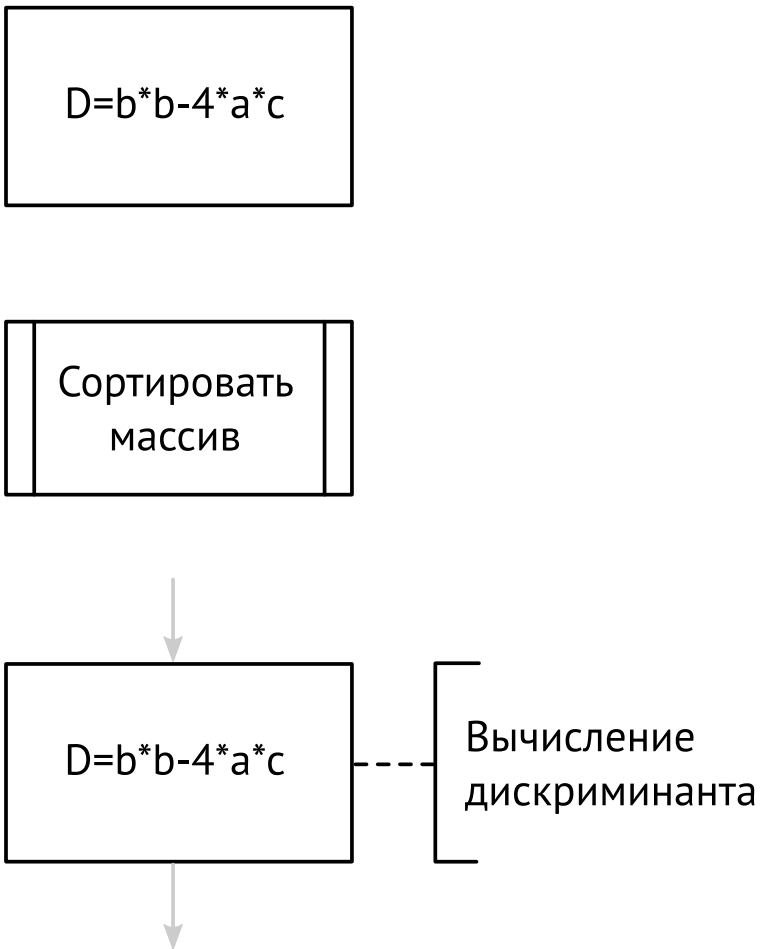
Блок-схема: решение квадратного уравнения



Блок-схема

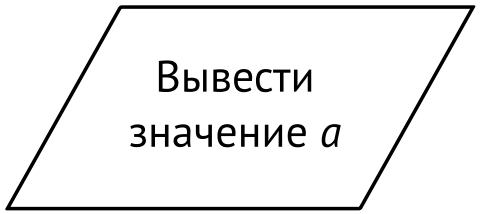
- Блок-схема — схема (графическая модель), описывающая алгоритм или процесс
- Отдельные шаги изображаются в виде блоков различной формы, соединенных между собой линиями, указывающими направление последовательности[8]
- Правила выполнения регламентируются ГОСТ 19.701-90 “Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения”

Процесс[8]



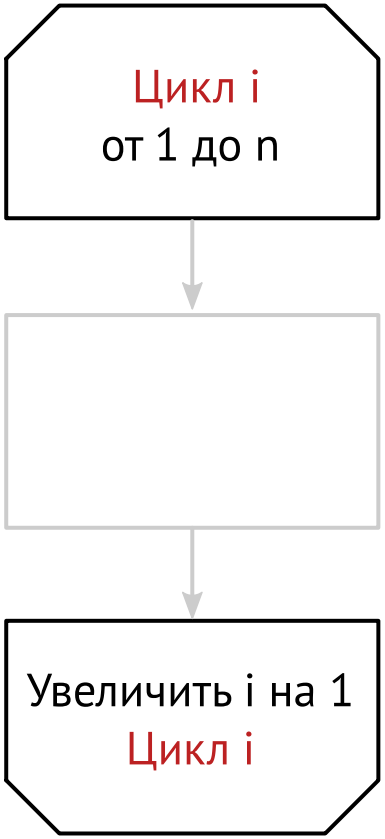
- Функция обработки данных любого вида
- Предопределённый процесс (вызов подпрограммы)
- Комментарии к процессу

Данные[8]



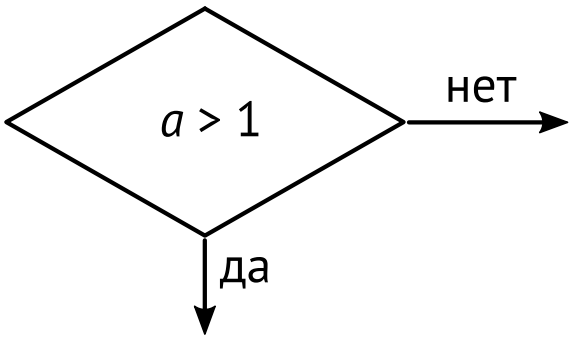
- Ввод данных
- Отображение результата

Цикл[8]



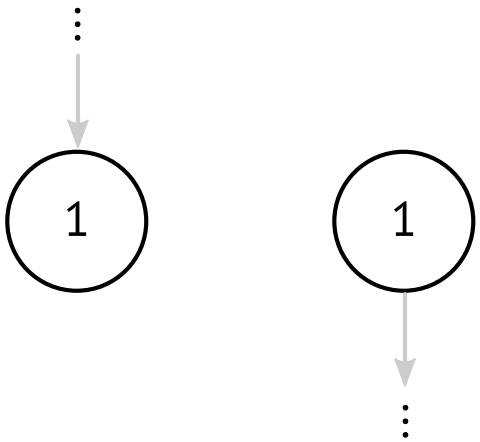
- Начало и конец цикла
- Обе части символа имеют одинаковый идентификатор

Решение[8]



- Решение или функция “переключательного” типа, имеющая один вход и ряд альтернативных выходов
- Только один выход может быть активизирован после вычисления условий

Соединитель [8]



- Символ отображает выход в часть схемы и вход из другой части этой схемы
- Используется для обрыва линии и продолжения ее в другом месте
- Соответствующие символы-соединители должны содержать одно и то же уникальное обозначение

Программа на языке Python

```
1  import math
2
3  print("Введите коэффициенты для уравнения (ax^2 + bx + c = 0):")
4
5  a = float(input("a="))
6  b = float(input("b="))
7  c = float(input("c="))
8
9  discr=b**2-4*a*c
10
11 if discr>0:
12    x1 = (-b+math.sqrt(discr))/(2*a)
13    x2 = (-b-math.sqrt(discr))/(2*a)
14    print('x1 = {:5.2f}, x2 = {:5.2f}'.format(x1, x2))
15 else:
16    print('Корней нет')
17
```

http://younglinux.info/python/task/quadratic

Этапы программирования

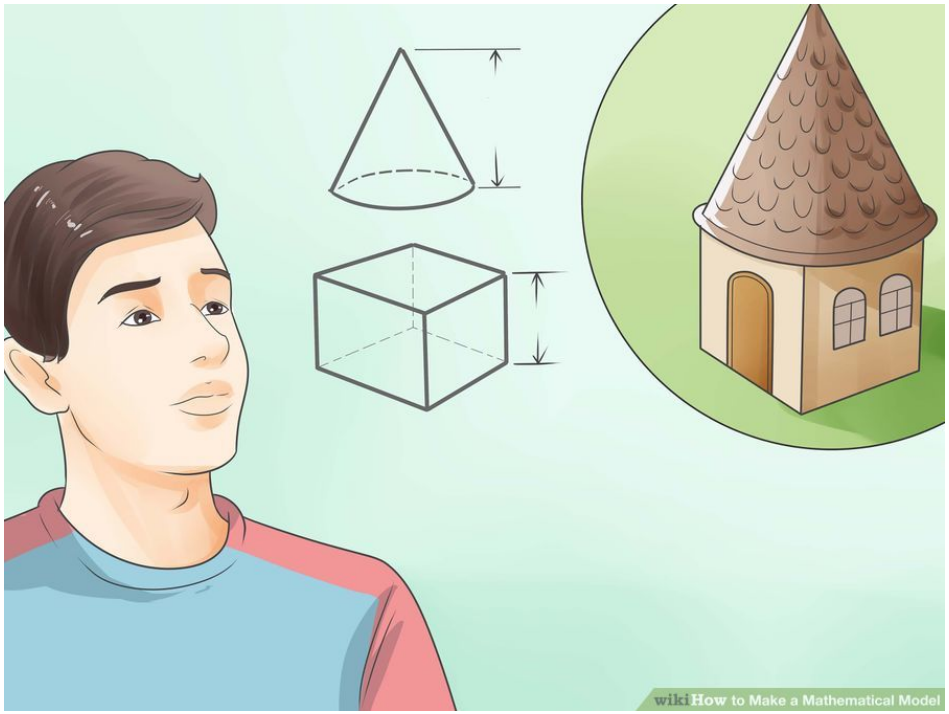
Этапы программирования[3]

- Постановка задачи на естественном языке
- Анализ задачи и моделирование
- Разработка алгоритма
- Программирование (кодирование)
- Тестирование и отладка
- Сопровождение

Постановка задачи на естественном языке

- сбор информации о задаче;
- формулировка условия задачи;
- определение конечных целей решения задачи;
- определение формы выдачи результатов;
- описание данных (типы, диапазоны, ...).

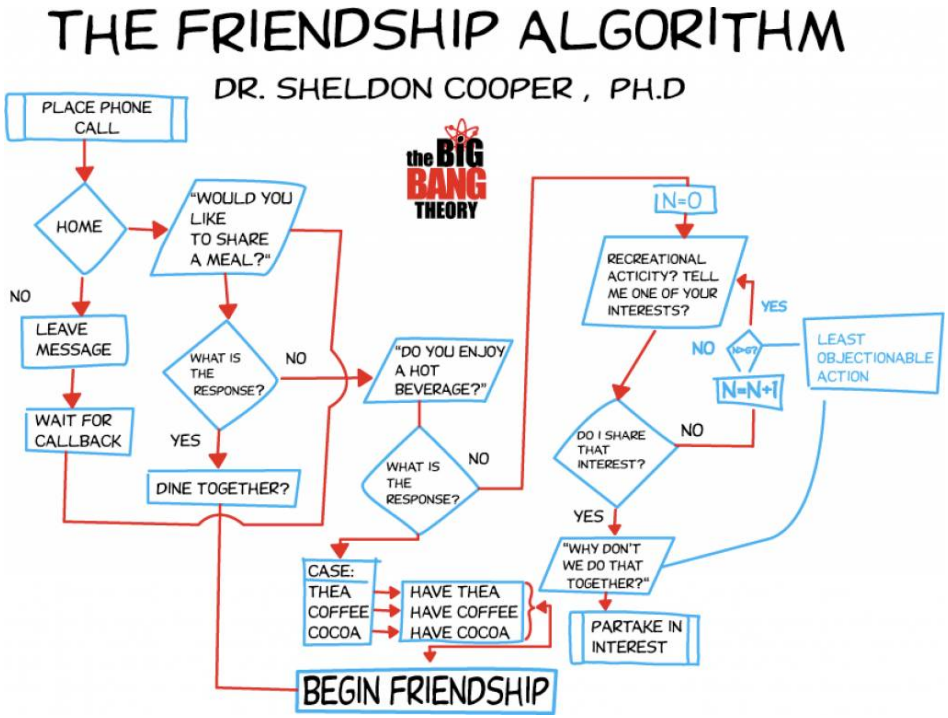
Анализ задачи и моделирование



Источник: www.wikihow.com

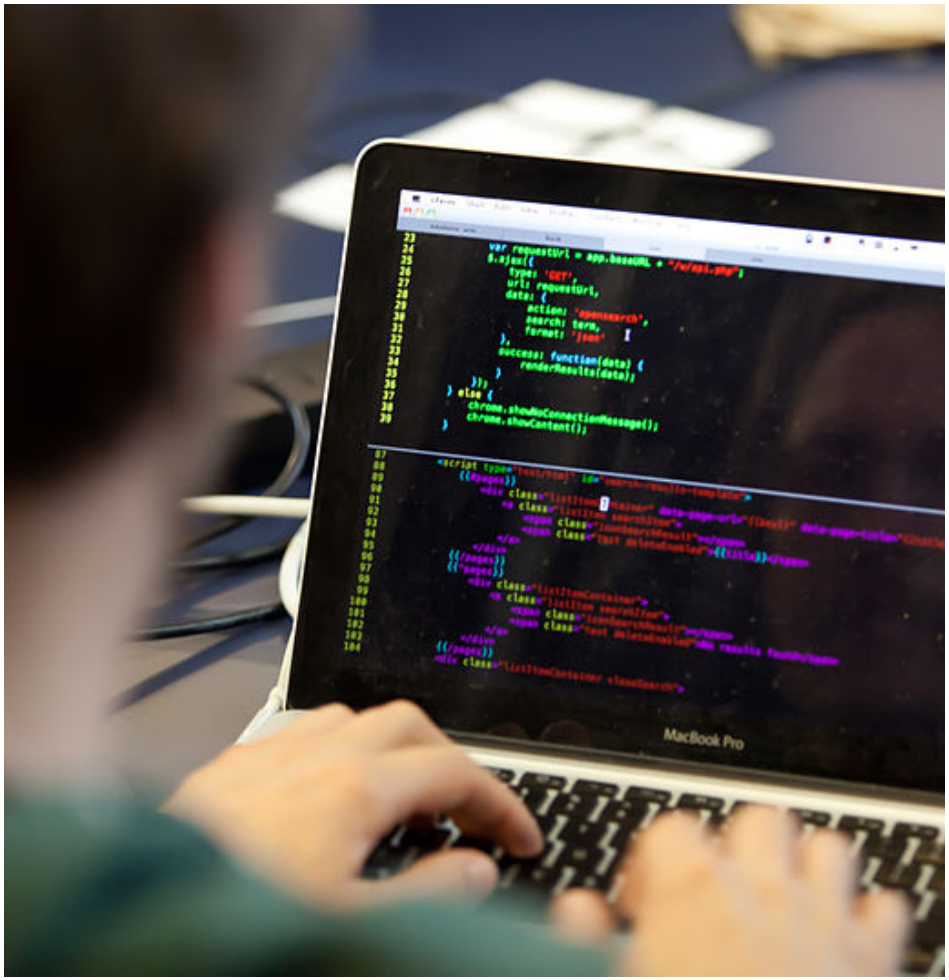
- анализ существующих аналогов;
- анализ технических и программных средств;
- разработка математической модели;
- разработка структур данных.

Разработка алгоритма



- выбор метода проектирования алгоритма;
- выбор формы записи алгоритма (блок-схемы, псевдокод и др.);
- выбор тестов и метода тестирования;
- проектирование алгоритма.

Программирование



- выбор языка программирования;
- уточнение способов организации данных;
- запись алгоритма на выбранном языке программирования.

Источник: https://commons.wikimedia.org

Тестирование и отладка

- синтаксическая отладка;
- отладка семантики и логической структуры;
- тестовые расчеты и анализ результатов тестирования;
- анализ результатов решения задачи, уточнение математической модели, алгоритма, программы.

Источник: https://www.pixafy.com

Сопровождение программы

- составление документации к решенной задаче, математической модели, алгоритму, программе, набору тестов, использованию;
- доработка программы для решения конкретных задач.

Источник: https://github.com/jtleek/rpackages

Программа

Программа

Данные, предназначенные для управления конкретными компонентами системы обработки информации в целях реализации определенного алгоритма[1]

Программное обеспечение

Совокупность программ системы обработки информации и программных документов, необходимых для эксплуатации этих программ

Программирование

Программирование требует знания

- предметной области для которой пишется программа,
- формальной логики,
- специализированных алгоритмов.

Язык программирования

Языки программирования

Язык программирования

формальная знаковая система, при помощи которой записываются компьютерные программы [2]

- Для языков программирования однозначно определены синтаксис и семантика.
- Описание синтаксиса языка включает определение алфавита и правил построения различных конструкций языка из символов алфавита.

Литералы. Идентификаторы. Ключевые слова

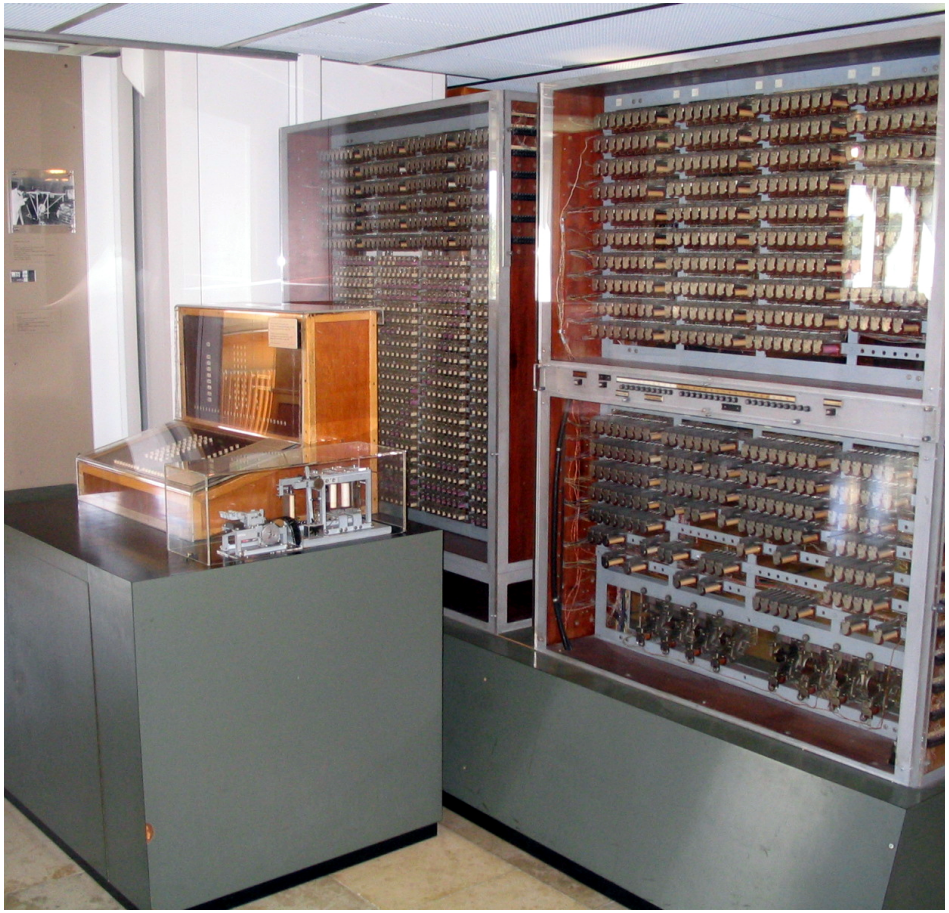
- Символы алфавита языка образуют **лексемы** — минимальные единицы языка, имеющие самостоятельный смысл[10]
- **Лексемы** формируют базовый словарь языка, понятный транслятору
  - ключевые слова: `False, class, return, is, else, ...`
  - идентификаторы – имена переменных, констант, массивов, функций
  - литералы – неизменяемые величины (строки текста, числа)
  - операции: `+ - * ** & < > <= ...`
  - знаки пунктуации (разделители): `( ) [ ] , : .`

Классификация языков программирования

- Машинно-зависимые языки
- Машинно-независимые языки

Машинно-зависимые языки

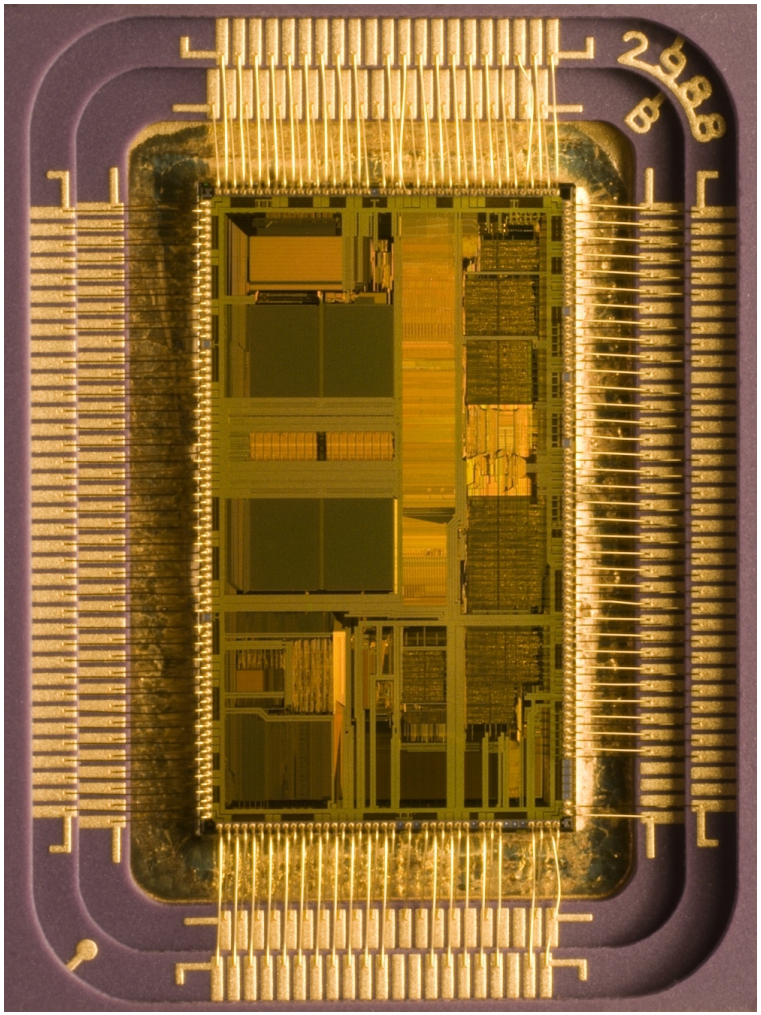
Z3 – Первая вычислительная машина



Источник: Wikimedia.org

- Конрад Цузе, 1941 г, Германия
- Первая программируемая в двоичном коде с плавающей точкой рабочая вычислительная машина
- Машина построена на 2600 телефонных реле
- Для хранения программ использовалась перфорированная лента
- Язык программирования Plankalkul (Исчисление планов)

Машинные коды



- Память ЭВМ может хранить лишь **двоичные представления**
- Программы для первых ЭВМ вводились в память с пульта, перемычками или тумблерами
- Программы писали в **машинных кодах**
- **Машинные коды** — **первое поколение** языков программирования, машинно-зависимых языков

Источник: Wikimedia.org

Машинный код

```
BB 11 01 B9 0D 00 B4
0E 8A 07 43 CD 10 E2
F9 CD 20 48 65 6C 6C
6F 2C 20 57 6F 72 6C
64 21
```

Код программы
“Hello World!”

- Машинный код – система команд (набор кодов операций) конкретной вычислительной машины, которая интерпретируется непосредственно процессором этой вычислительной машины
- Машинный код может исполнятся ещё более низкоуровневым слоем программ (процессора), называемых микропрограммами

Машинно-зависимые языки программирования

- Языки программирования, учитывающий структуру и характеристики ЭВМ определенного типа
- Имеют непосредственный доступ к аппаратным средствам ЭВМ

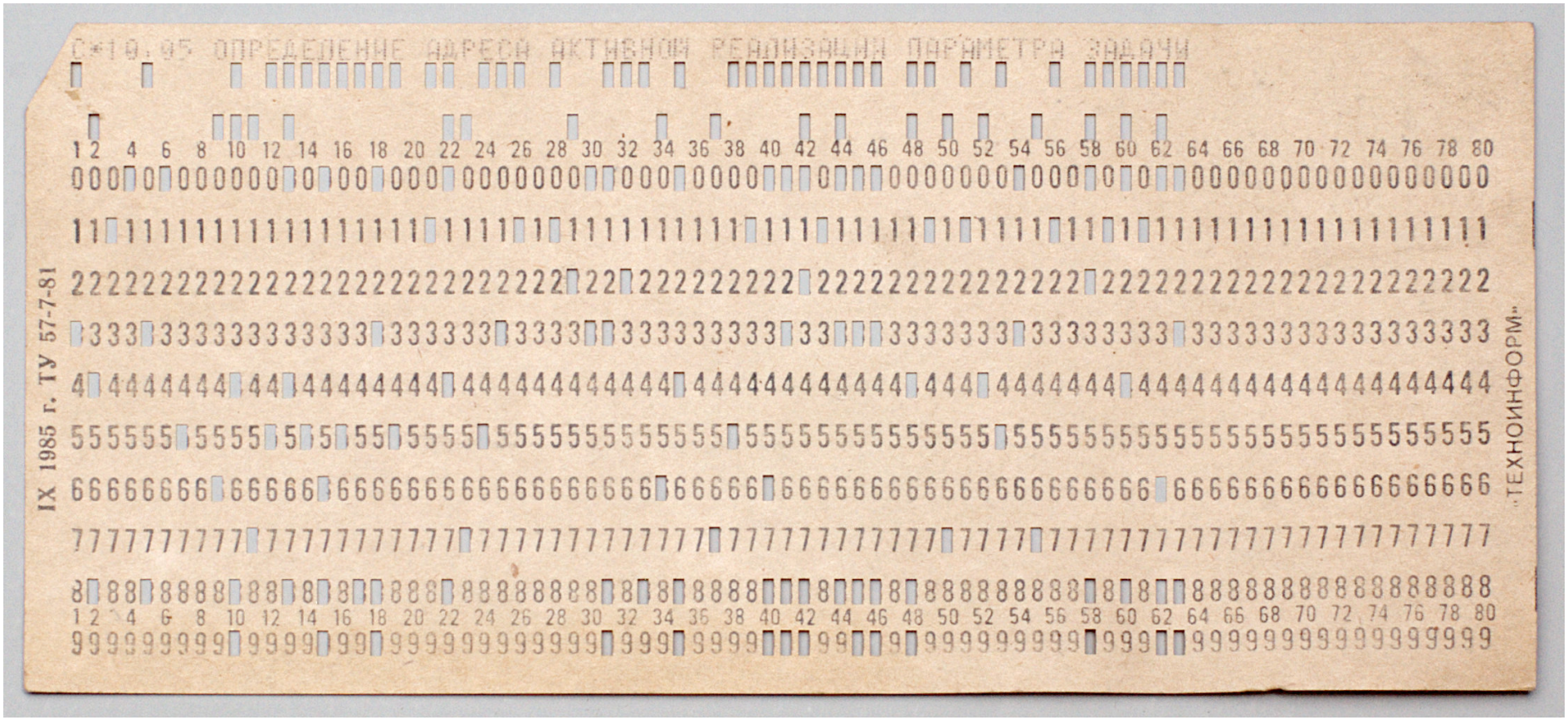
Машинно-ориентинованные языки[6]

Мнемокод (язык символического кодирования, язык ассемблера)

- Коды операций, адреса ячеек памяти заменяются буквами или буквенно-цифровыми обозначениями
- **Программа-транслятор (ассемблер)** переводит программу на языке ассемблера в исполнимый машинный код
- Применяется для написания программ, явно использующих специфику конкретной аппаратуры

Язык ассемблера

```
1  .MODEL SMALL
2  .DATA
3      msg DB 'Hello World',13,10,'$'
4  .CODE
5  START:
6      mov ax, @DATA
7      mov ds, ax
8      mov ax, 0900h
9      lea dx, msg
10     int 21h
11     mov ax, 4C00h
12     int 21h
13 END START
```


C*10.95 ОПРЕДЕЛЕНИЕ АДРЕСА АКТИВНОЙ РЕАЛИЗАЦИИ ПАРАМЕТРА ЗАДАЧИ
IX 1985 г. ТУ 57-7-81
"ТЕХНОИНФОРМ"

Машинно-ориентированные языки[6]

Макроязык

- Содержит макрокоманды, не имеющие аналогов в машинном языке
- При переводе на машинный язык каждая макрокоманда заменяется группой команд машинного языка

- Ассемблер относится к языкам **второго поколения**

Язык программируемого калькулятора МК-61

Программа “Лунолёт-1”

	0	1	2	3	4	5	6	7	8	9
0	Пх1	С/П	Пх3	С/П	Пх2	С/П	хП4	1	0	x
10	Пх5	÷	Пх6	–	хП7	Пх8	x	Пх3	+	хП9
20	Fx^2	В↑	Пх3	Fx^2	–	Пх7	÷	2	÷	Пх1
30	+	хП1	Пх9	хП3	Пх2	Пх4	Пх8	–	хП2	K|x|
40	2	–	Fx<o	0	Пх3	K|x|	5	–	Fx≥o	54
50	6	6	6	С/П	7	7	7	С/П		

Цель игры: посадить корабль на поверхность Луны со скоростью не больше 5 м/с. Управление осуществляется расходом топлива в единицу времени, определяющим силу тяги двигателя. Количество топлива ограничено.

Машинно-независимые языки

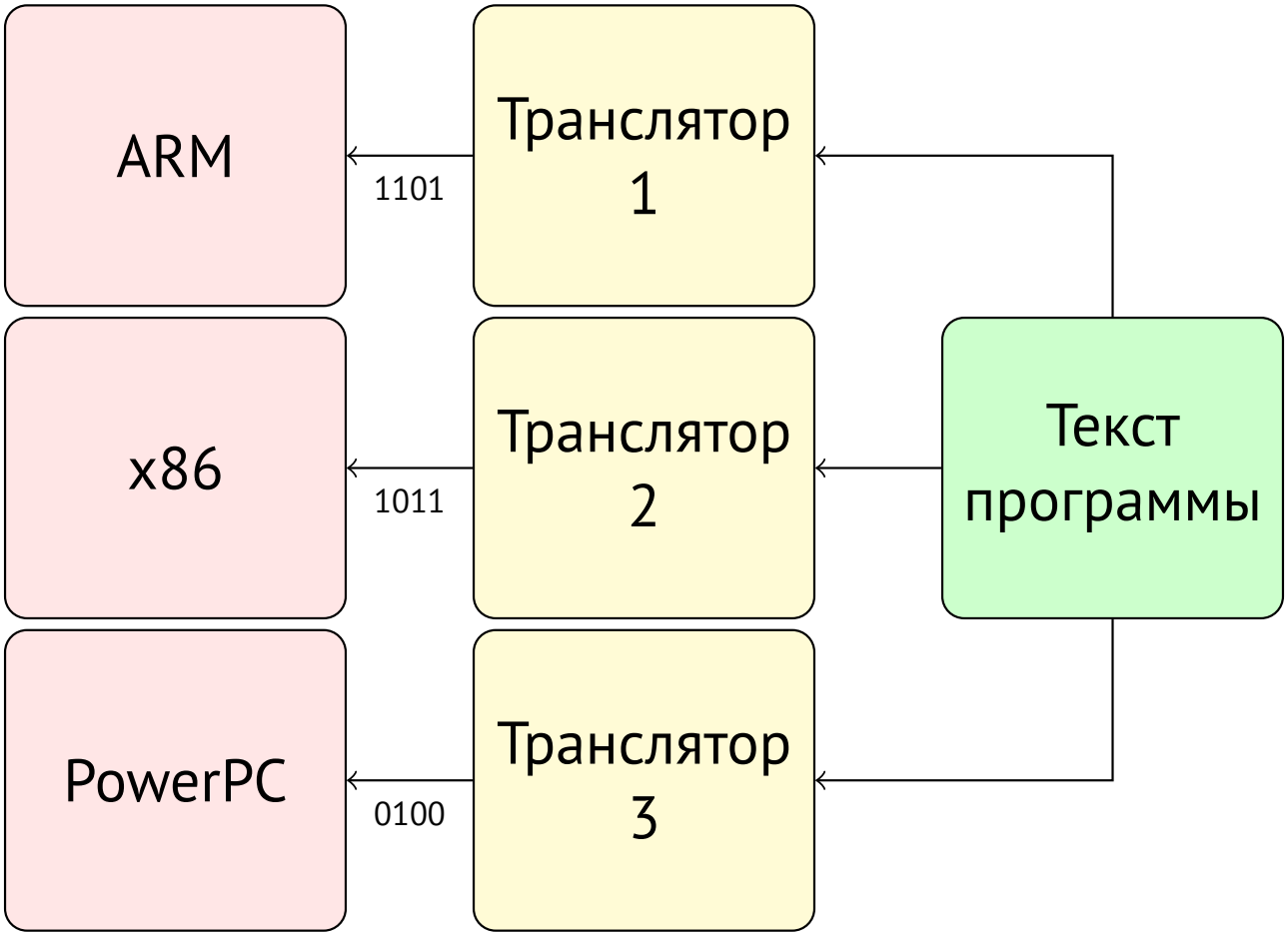
Машинно-независимые языки программирования

Языки программирования, структура и средства которого не связаны с конкретной ЭВМ и позволяют выполнять составленные на нем программы на любой ЭВМ, снабженной **трансляторами** с этого языка.

Трансляция

- Программа, написанная на языке высокого уровня преобразуется (транслируется) в машинный код при помощи специальной программы — **транслятора**
- **Транслятор** — это специальная программа, преобразующая текстовый файл программы в исполняемый код

Трансляция для разных платформ



Компиляторы и интерпретаторы

Для преобразования команд языка программирования в машинный код используются трансляторы, которые бывают двух типов

- Интерпретаторы
- Компиляторы

Компиляторы

- **Компилятор** — программа или техническое средство, выполняющее **компиляцию**.
- **Компиляция** — трансляция (перевод) программы, составленной языке высокого уровня, в эквивалентную программу на низкоуровневом языке, близком машинному коду.
- Результат компиляции — исполнимый модуль обладает максимальной возможной производительностью и привязан к определённой операционной системе и процессору.

Преимущества и недостатки компиляции

- ✓ Высокая скорость (исполняется машинный код)
- ✓ Не требуется дополнительного ПО для запуска* программы
- ✗ Время отладки программы увеличивается (после исправления ошибки необходима перекомпиляция всей программы)
- ✗ Программу нельзя запустить на ЭВМ другого типа

Интерпретатор

Интерпретатор выполняет построчный анализ, обработка и выполнение программы, без обработки всего текста программы

Преимущества и недостатки интерпретации

Преимущества и недостатки

- ✓ Программа может исполняться на разных платформах
- ✓ Меньший размер кода (нет машинного кода, есть только текст)
- ✓ Простая отладка программы
- ✗ Невозможно выполнение программы без интерпретатора
- ✗ Медленное исполнение (затраты времени на анализ выражений и невозможность оптимизации)

Трансляция Си в код ассемблера

```c
#include <stdio.h>

int main (void)
{
 puts ("Hello, World!");
 return 0;
}
```

```
gcc -O2 -S -c hworld.c
```

```
        .file   "hworld.c"
        .section        .rodata.str1.1,"aMS",@progbits,1
.LC0:
        .string "Hello, World!"
        .section        .text.unlikely,"ax",@progbits
.LCOLDB1:
        .section        .text.startup,"ax",@progbits
.LHOTB1:
        .p2align 4,,15
        .globl  main
        .type   main, @function
main:
.LFB23:
        .cfi_startproc
        subq    $8, %rsp
        .cfi_def_cfa_offset 16
        movl    $.LC0, %edi
        call    puts
        xorl    %eax, %eax
        addq    $8, %rsp
        .cfi_def_cfa_offset 8
        ret
        .cfi_endproc
.LFE23:
        .size   main, .-main
        .section        .text.unlikely
.LCOLDE1:
        .section        .text.startup
.LHOTE1:
        .ident  "GCC: (Ubuntu 5.4.0) 5.4.0 20160609"
        .section        .note.GNU-stack,"",@progbits
```

Трансляция Си в машинный код

```c
#include <stdio.h>

int main (void)
{
 puts ("Hello, World!");
 return 0;
}
```

```
gcc -Os hworld.c -o hworld
```

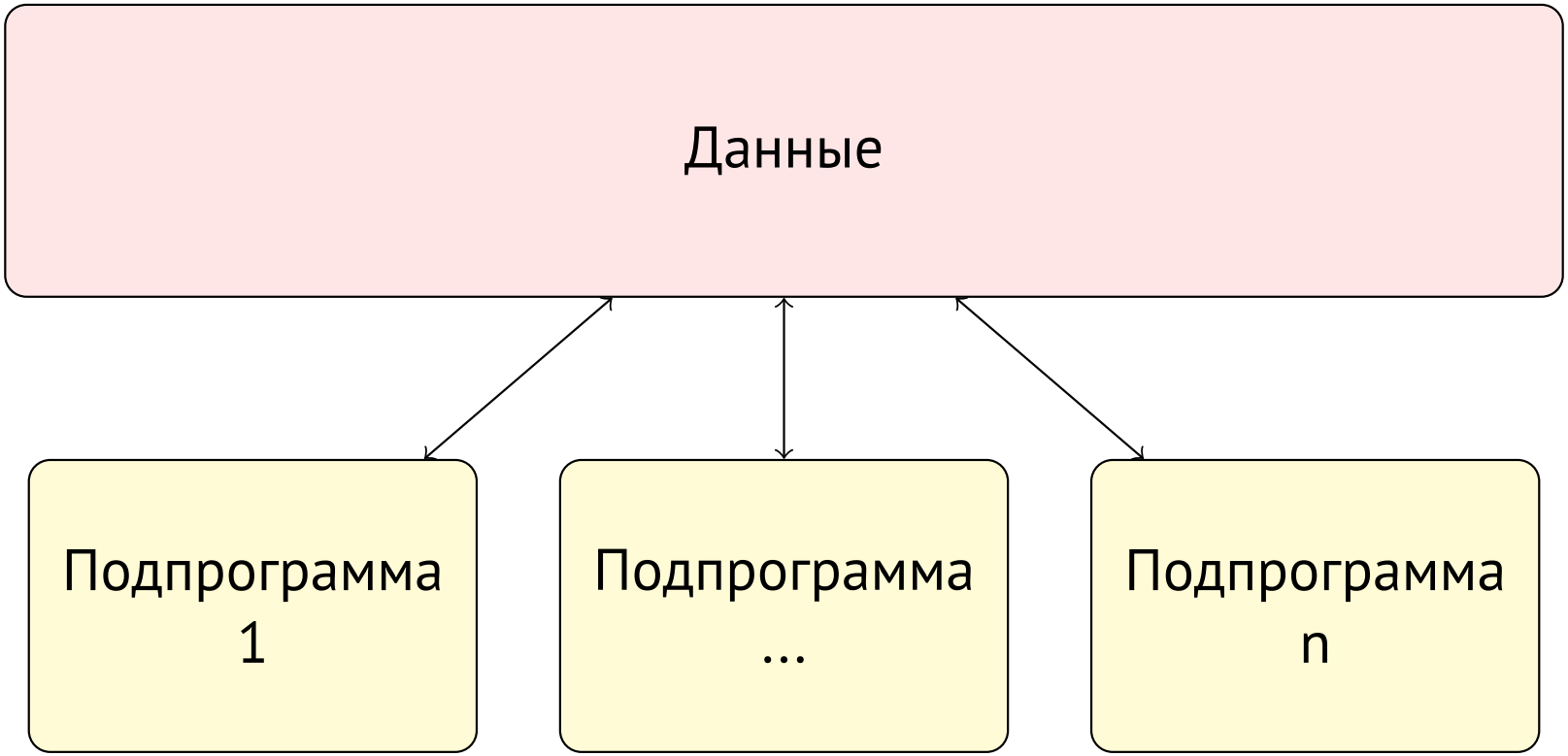
10% машинного кода

```
0000000 457f 464c 0102 0001 0000 0000 0000 0000
0000010 0002 003e 0001 0000 0440 0040 0000 0000
0000020 0040 0000 0000 0000 19e0 0000 0000 0000
0000030 0000 0000 0040 0038 0009 0040 001f 001c
0000040 0006 0000 0005 0000 0040 0000 0000 0000
0000050 0040 0040 0000 0000 0040 0040 0000 0000
0000060 01f8 0000 0000 0000 01f8 0000 0000 0000
0000070 0008 0000 0000 0000 0003 0000 0004 0000
0000080 0238 0000 0000 0000 0238 0040 0000 0000
0000090 0238 0040 0000 0000 001c 0000 0000 0000
00000a0 001c 0000 0000 0000 0001 0000 0000 0000
00000b0 0001 0000 0005 0000 0000 0000 0000 0000
00000c0 0000 0040 0000 0000 0000 0040 0000 0000
00000d0 06f4 0000 0000 0000 06f4 0000 0000 0000
00000e0 0000 0020 0000 0000 0001 0000 0006 0000
00000f0 0e10 0000 0000 0000 0e10 0060 0000 0000
0000100 0e10 0060 0000 0000 0228 0000 0000 0000
0000110 0230 0000 0000 0000 0000 0020 0000 0000
0000120 0002 0000 0006 0000 0e28 0000 0000 0000
0000130 0e28 0060 0000 0000 0e28 0060 0000 0000
0000140 01d0 0000 0000 0000 01d0 0000 0000 0000
0000150 0008 0000 0000 0000 0004 0000 0004 0000
0000160 0254 0000 0000 0000 0254 0040 0000 0000
0000170 0254 0040 0000 0000 0044 0000 0000 0000
0000180 0044 0000 0000 0000 0004 0000 0000 0000
0000190 e550 6474 0004 0000 05d4 0000 0000 0000
00001a0 05d4 0040 0000 0000 05d4 0040 0000 0000
00001b0 0034 0000 0000 0000 0034 0000 0000 0000
00001c0 0004 0000 0000 0000 e551 6474 0006 0000
00001d0 0000 0000 0000 0000 0000 0000 0000 0000
```

Языки, используемые при создании программ

Поколения языков высокого уровня

Языки первого поколения[11]



Языки первого поколения

- Программы состоят из подпрограмм
- Используются глобальные переменные — данные открытые для всех подпрограмм
- ✗ Ошибка, допущенная в одной части программы, может оказать разрушительное влияние на остальную часть системы
- ✗ Низкая надежность, запутанность большой программы

ФОРТРАН (FORTRAN — FORmula TRANslator)

- Разработан в 1954-1957 годах программистами под руководством Джона Бэкуса (John W. Backus) в IBM
- Разработан и используется по настоящее время для научных и инженерных вычислений (BLAS, LINPACK, EISPACK, ODEPACK, IMSL)
- Для эффективности вычислений многие конструкции языка первоначально разрабатывались с учетом архитектуры машины IBM 407
- Существуют стандарты языка: FORTRAN 66, FORTRAN 90, FORTRAN 95, FORTRAN 2003, FORTRAN 2008

Алгол

- Разработан в 1958-1960 годах как универсальный язык программирования
- Авторы языка хотели создать язык независимый от конкретной архитектуры вычислительной системы, удобный для описания алгоритмов и применяющий систему обозначений, близкую к той, что принята в математике.
- Широко использовался в Европе и СССР для научных вычислений
- Появилась блочная структура организации программ: каждый блок (главная программа, подпрограммы) ограничивался ключевыми словами begin и end
- Появились рекурсивные процедуры

Программа на языке Алгол-60

Дана исходная система семи дифференциальных уравнений первого порядка:

1). $\frac{dt}{dt} = 1$;

2). $\frac{dx}{dt} = \dot{x}$;

3). $\frac{dy}{dt} = \dot{y}$;

4). $\frac{d\vartheta}{dt} = \dot{\vartheta}$;

5). $\frac{d\dot{x}}{dt} = \frac{(P_1 - C_\tau \frac{\rho v^2}{2} S_м)\cos\vartheta + (P_2 \mp C_N \frac{\rho v^2}{2} S_м)\sin\vartheta}{m}$;

6). $\frac{d\dot{y}}{dt} = \frac{(P_1 - C_\tau \frac{\rho v^2}{2} S_м)\sin\vartheta - (P_2 \mp C_N \frac{\rho v^2}{2} S_м)\cos\vartheta}{m} - g$;

7). $\frac{d\dot{\vartheta}}{dt} = \frac{-P_2(x_р - x_т) \mp C_N \frac{\rho v^2}{2} S_м (x_т - x_д)}{J}$;

где: $P_1, P_2, \rho, S_м, m, g, x_р, x_т, x_д, J$ - постоянные;
$C_N, C_\tau, v, \vartheta$ - переменные величины.

Требуется определить: $x, y, \vartheta, \dot{x}, \dot{y}, \dot{\vartheta}$.

Программа составлена на языке АЛГОЛ-60.

Исходные данные заданы массивом $isx[1:21]$, величины C_N и C_τ заданы массивами $mcn, mct[1:12]$. Исходная система дифференциальных уравнений решается методом Рунге-Кутта с постоянным шагом интегрирования.

Выход из решения осуществляется по одной функции: $f = \dot{y}$.

На печать выдается на первом шаге и, затем, через каждые 10 шагов: $t, x, y, \vartheta, \dot{x}, \dot{y}, \dot{\vartheta}$.

В программе используются следующие стандартные программы:

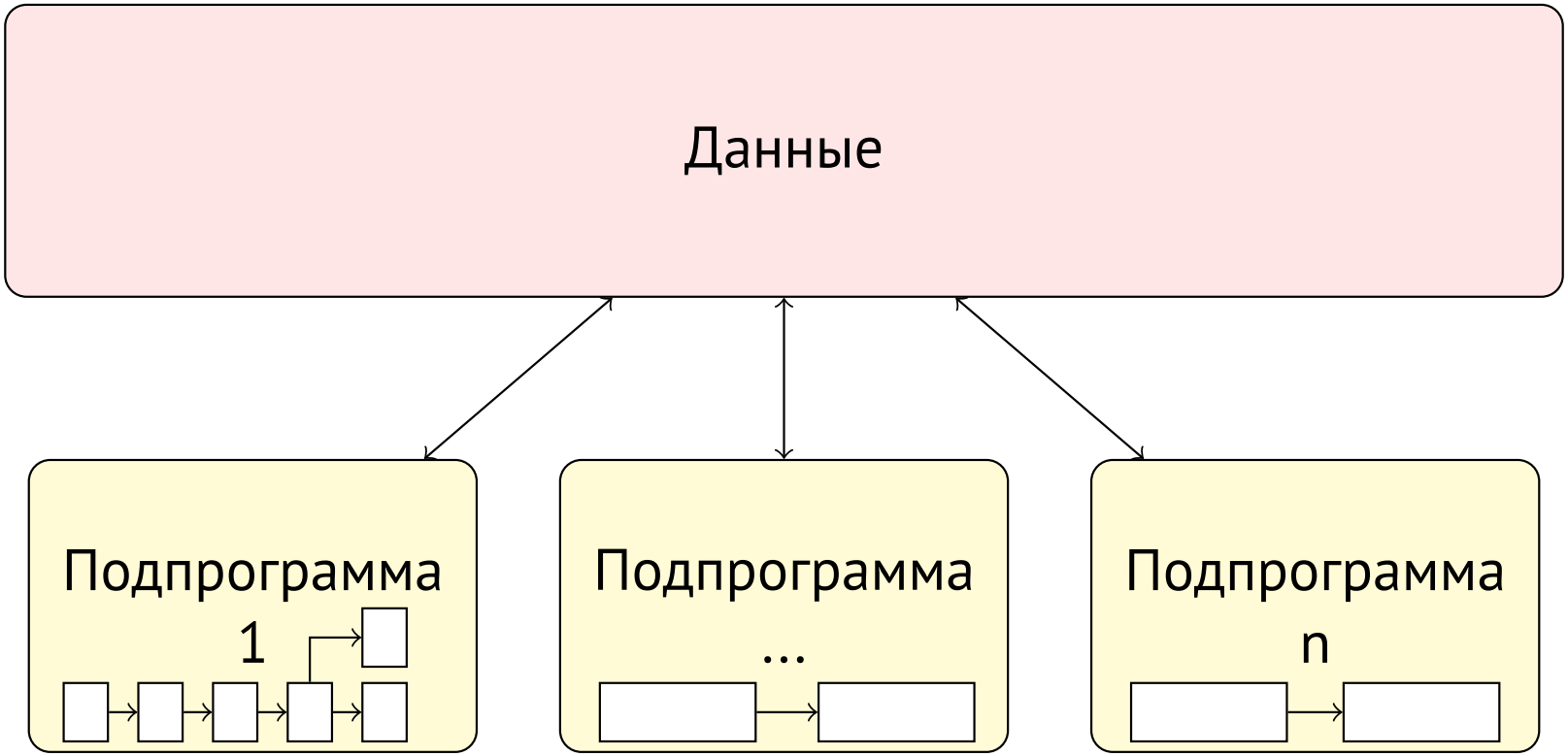
0042
I045
I04I
I07I

Необходимо расширение рабочего поля ИС-2. В данной программе РП с 6000 ячейки. Программный останов: 0077 в ячейке I242. Время счета - 5 минут.

Дата составления программы: I7.Ш.67.

```
Begin array isx[1:21], к[1:28], e[1:3], τ[1:4],
mct, mcn[1:12], ct, cn[1:2], f[1:7];
Boolean array pi[1:3];
real δк, s, u, a, p1, p2, p1x, p1y, p2x, p2y, n, mn;
integer i, j, h;
P0042(isx, mct, mcn);
for i:=1 step 1 until 7 do к[i]:=isx[i+14];
e[1]:=isx[2]; e[2]:=0; j:=isx[14]-1;
P1045(isx[1], 7, 1, к, 0, e, true, AN, AK, AK, BK);
go to BK;
AN: δк:=к[12]↑2+к[13]↑2; s:=isx[6]×δк×isx[5]/2;
u:=arctg(к[13]/к[12])×57.3; a:=к[11]-u;
ct[1]:=abs(a); P1071(6,1,mct,ct,τ,1,1,false);
if к[8]<isx[13] then h:=1 else h:=0;
p1:=isx[8]×h-ct[2]×s; p1x:=p1×cos(к[11]/57.3);
p1y:=p1×sin(к[11]/57.3); cn[1]:=abs(a);
P1071(6,1,mcn,cn,τ,1,1,false); n:=cn[2]×s;
if a>0 then begin mn:=-n; p2:=isx[9]×h-n;
go to C end else mn:=n; p2:=isx[9]×h+n;
C: p2x:=p2×sin(к[11]/57.3); p2y:=p2×cos(к[11]/57.3);
к[8]:=1; for i:=1 step 1 until 3 do к[i+8]:=к[i+11];
к[12]:=(p1x+p2x)/isx[3]; к[13]:=(p1y-p2y)/isx[3]-isx[7];
к[14]:=57.3×(-isx[9]×h×(isx[10]-isx[12])+mn×
(isx[12]-isx[11]))/isx[4];
AK: e[3]:=к[6]; j:=j+1; if j=isx[14] then begin
for i:=1 step 1 until 7 do f[i]:=к[i]; j:=0;
P1041(f) end; BK: stop end
```

Структурное программирование



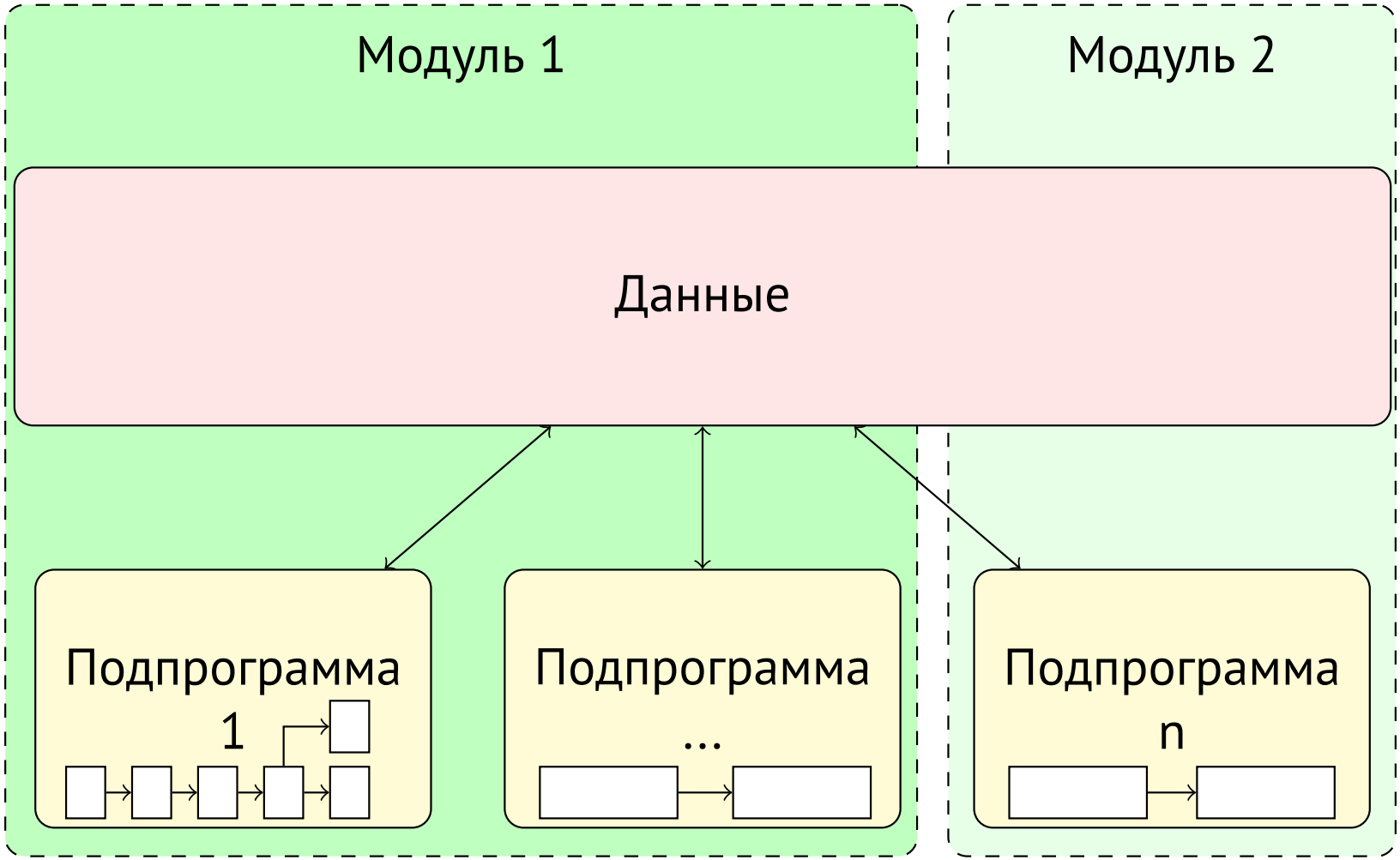
Структурное программирование

- Программа разбивается на составляющие элементы
- В программе четко обозначенны управляющие структуры, программные блоки, автономные **подпрограммы**.
- Базовые управляющие структуры[4]
  - последовательность
  - ветвление
  - цикл
- Используются **локальные переменные**[9]
- Разработка программы ведётся пошагово, методом “сверху вниз”

Язык Паскаль

```
1  program QuadraticEquation;
2  var
3    a,b,c,d,x1,x2: real;
4  begin
5    write('введите коэффициенты уравнения a,b,c');
6    readln(a,b,c);
7    d:= b*b-4*a*c;
8    if d >= 0 then begin
9      x1:= (-b+sqrt(d))/(2*a);
10     x2:= (-b-sqrt(d))/(2*a);
11     writeln('x1=',x1:4:2, ', x2=',x2:4:2);
12   end
13   else begin
14     writeln('Действительных корней нет');
15   end
16 end.
```

Топология языков третьего поколения



Модульное программирование

- Программа разбивается на модули изолированные программные сегменты с четко определённым **интерфейсом**, описывающим как подготовить данные и как вызвать функции модуля преобразующие эти данные.
- Отладка модулей может проводиться независимо
- Модули одной программы могут разрабатываться на различных языках программирования
- Модули могут использоваться повторно

Объектно-ориентированное программирование

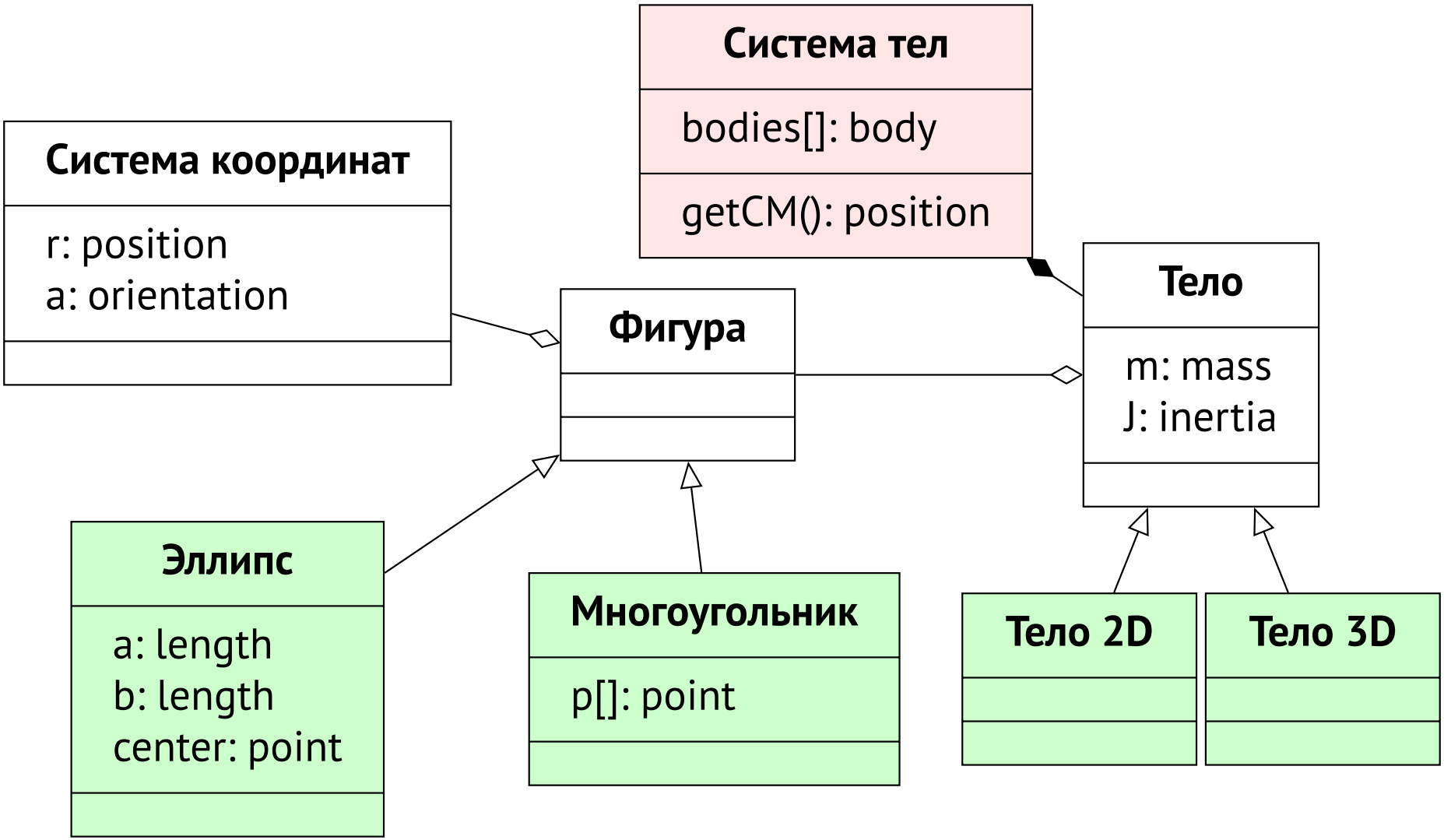
Основные принципы

- Основной элемент конструкции — модуль
- Модуль состоит из связанных классов и объектов
- Классы образуют иерархию (наследование, включение)

Объектные и объектно-ориентированные языки

- Первый язык с поддержкой объектно-ориентированного программирования Simula-67
- Smalltalk (70е годы) — первый объектно-ориентированный язык программирования

Иерархия классов



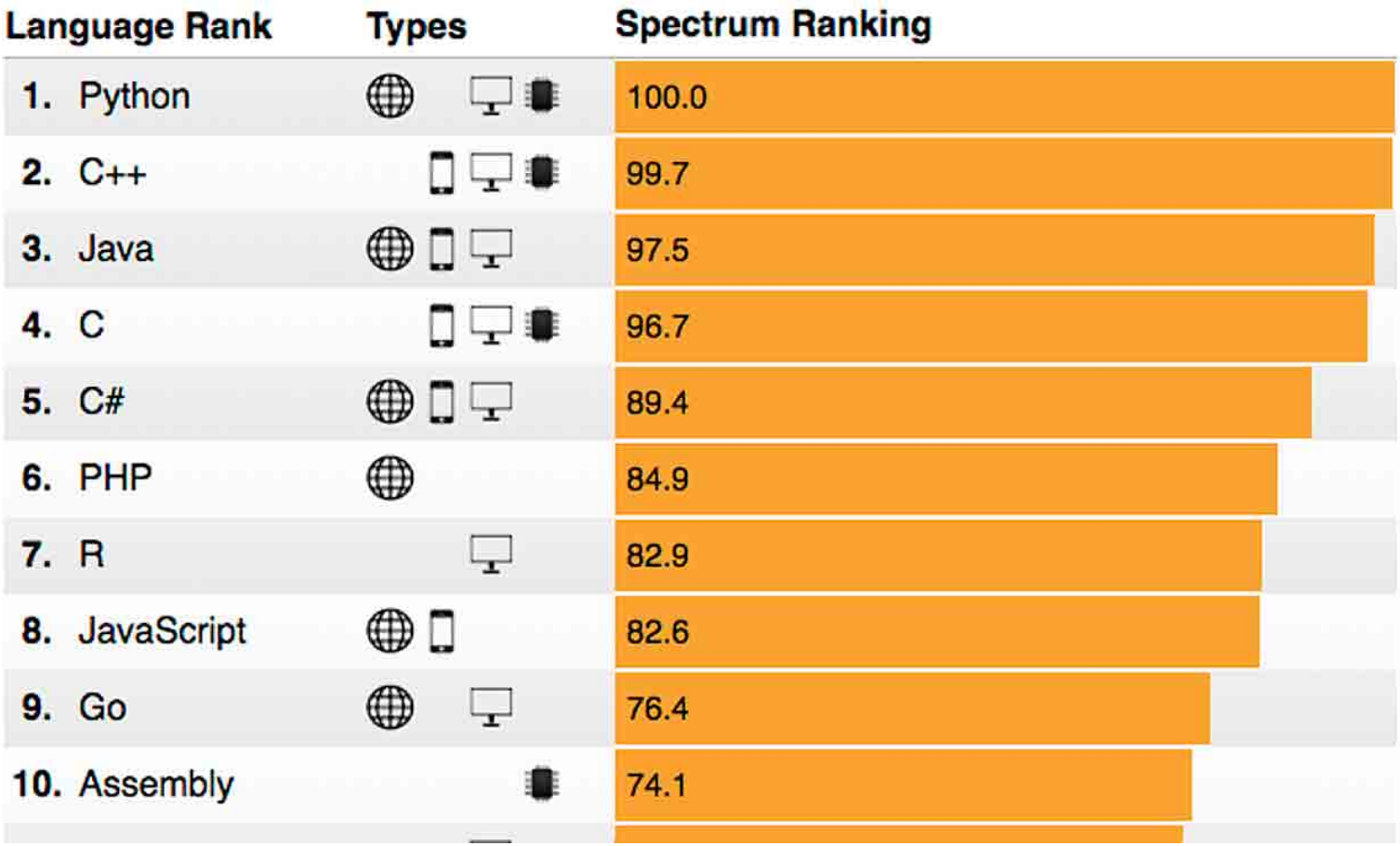
Выбор языка программирования

- Выбор языка зависит от решаемой задачи
- Разные языки требуют от программиста различного уровня внимания к деталям при реализации алгоритма

Области применения

Научные вычисления	Python, C++, FORTRAN, Java
Системное программирование	C, C++, Java
Искусственный интеллект	LISP, Prolog
Издательская деятельность	Postscript, TeX, LaTeX
Описание документов	HTML, XML

Языки программирования



Источник: http://spectrum.ieee.org/static/interactive-the-top-programming-languages-2018

Список источников

Список источников

[1] ГОСТ 19781—90 Обеспечение систем обработки информации программное.

[2] Progopedia, “Энциклопедия языков программирования.” [Online]. Available: http://progopedia.ru/. [Accessed: 04-Sep-2016].

[3] Бородина Т. А. Turbo Pascal для школьников [Online]. Available: http://tat67183862.narod.ru/ [Accessed: 04-Sep-2016].

[4] Wikipedia. Structured programming [Online]. Available: https://en.wikipedia.org/wiki/Structured_programming [Accessed: 04-Sep-2016].

[5] J. Daintith, “A dictionary of computing,” Oxford dictionary of computing, vol. 6. p. 567, 2008.

[6] А.Д.Липенков Информатика [Online]. Available: http://uchu2008.narod.ru/razdely/informatika/inform_lectures/soderganie/temy.html [Accessed: 04-Sep-2016].

[7] Wikipedia. LaTeX [Online]. Available: https://ru.wikipedia.org/wiki/LaTeX [Accessed: 04-Sep-2016].

[8] Wikipedia. Блок-схема [Online]. Available: https://ru.wikipedia.org/wiki/Блок-схема [Accessed: 04-Sep-2016].

[9] Игорь Дидковский, Перспективные ОС и языки порграммирования. Available: http://e-skin.hut.ru/langs/ [Accessed: 04-Sep-2016].

[10] INF-W Available: http://inf-w.ru/?page_id=4367 [Accessed: 04-Sep-2016].

[11] Г. Буч, Объектно-ориентированный анализ и проектирование с примерами приложений на C++. Второе издание. М.: “Издательство Бином”, 1998.