ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ имени И.С. Тургенева

Ц М И

cdio™

АСУиК Автоматизированные системы управления и кибернетика

Министерство образования и науки РФ

Алгоритмизация и программирование

Пилипенко Александр Витальевич

руководитель рабочей группы проекта, директор центра междисциплинарного инжиниринга, руководитель образовательных программ по направлениям «Управление в технических системах» и «Автоматизация технологических процессов и производств» ОГУ имени И.С. Тургенева

СКОЛКОВО
Московская школа управления

Алгоритм

Министерство образования и науки РФ

- Алгоритм – конечная последовательность точных предписаний (правил), однозначно определяющих процесс преобразования исходных и промежуточных данных в конечный результат.
- Алгоритм — это всякая система вычислений, выполняемых по строго определённым правилам, которая после какого-либо числа шагов заведомо приводит к решению поставленной задачи

СКОЛКОВО
Московская школа управления

Определения

Министерство образования и науки РФ

- Алгоритм — это последовательность действий, направленных на получение определённого результата за конечное число шагов.
- Алгоритм — это последовательность действий, которая ведёт к конечному результату

СКОЛКОВО
Московская школа управления

Свойства алгоритмов

Министерство образования и науки РФ

- ***Массовость***. алгоритм решения задачи разрабатывается в общем виде и должен быть применим для некоторого класса задач, различающихся только входными данными.
- ***Результативность***. При любых допустимых исходных данных алгоритм должен быть завершен с определёнными результатами.
- ***Понятность*** — алгоритм для исполнителя должен включать только те команды, которые ему (исполнителю) доступны, которые входят в его систему команд.

СКОЛКОВО
Московская школа управления

Свойства алгоритмов

Министерство образования и науки РФ

- ***Завершаемость (конечность)***. Алгоритм должен завершать работу и выдавать результат за конечное число шагов.
- ***Дискретность***. Рассматриваемы процесс может быть разделен на элементарные части.
- ***Детерминированность***. Каждый этап алгоритма должен быть сформулирован так, что бы предусматриваемые им действия определялись однозначно.

представления алгоритмов

СКОЛКОВО
Московская школа управления

Министерство образования и науки РФ

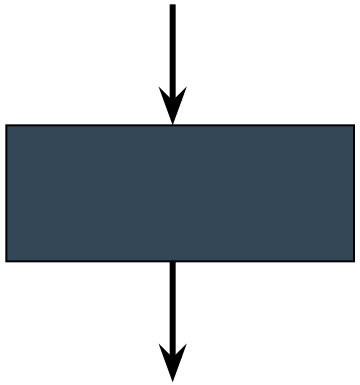
1. Словесная запись. Недостаток: отсутствие строгой формализации и наглядности вычислительного процесса.
2. Табличная форма записи алгоритма.
3. Логические схемы алгоритмов.
4. Графовые схемы алгоритмов. (Блок схемы)
5. Псевдокод

СКОЛКОВО
Московская школа управления

Блок-схемы

Министерство образования и науки РФ

- Блок-схема – графическое изображение программы дополненное элементами словесной записи. Каждый этап программы представляется определенной графической фигурой.

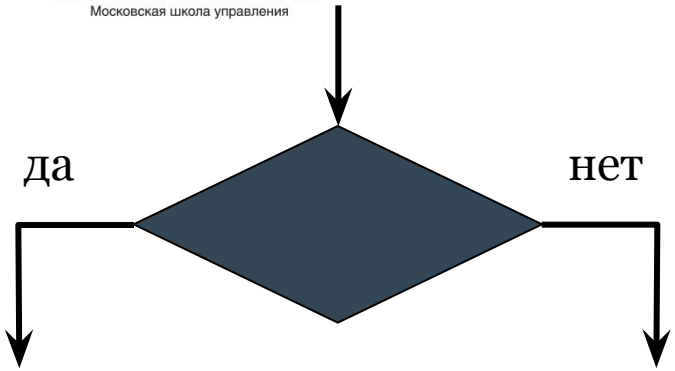


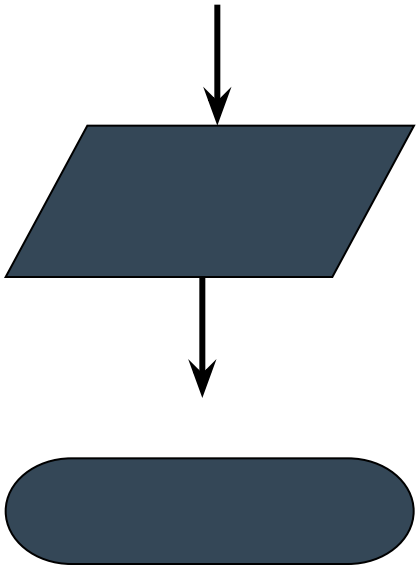
Выполнение операции или группы операций в результате, которого изменяется значение, форма представления или расположение данных

СКОЛКОВО
Московская школа управления

Министерство образования и науки РФ

Элементы блок-схем





- Условие. Выбирается дальнейшее направление выполнения программы в зависимости от выполнения условия
- Ввод и вывод данных из программы
- Начало и конец программы

СКОЛКОВО
Московская школа управления

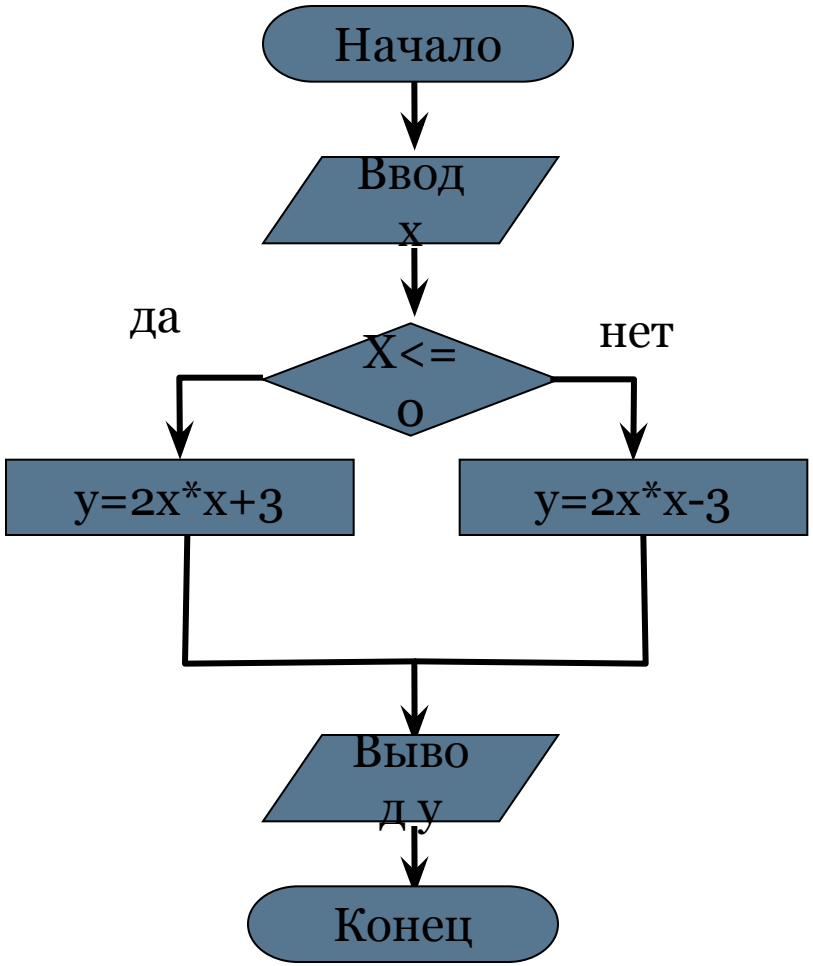
Пример 1

Министерство образования и науки РФ

Разветвляющийся алгоритм

Алгоритм называется разветвляющимся, если он содержит несколько ветвей выполнения программы отличающихся друг от друга содержанием вычислений

$$y = \begin{cases} 2x^2 + 3, & x \le 0 \\ 2x^2 - 3, & x > 0 \end{cases}$$



Пример 2

СКОЛКОВО
Московская школа управления

Министерство образования и науки РФ

Циклический алгоритм

Алгоритм называется циклическим, если он содержит многократное выполнение одних и тех же ветвей при различных значениях промежуточных данных

СКОЛКОВО
Московская школа управления

Псевдокод

Министерство образования и науки РФ

- Псевдокод — компактный язык описания алгоритмов, использующий ключевые слова языков программирования, но опускающий несущественные подробности и специфический синтаксис.
- Псевдокод обычно опускает детали, несущественные для понимания алгоритма человеком. Такими несущественными деталями могут быть описания переменных, системно-зависимый код и подпрограммы.

СКОЛКОВО
Московская школа управления

Пример псевдокода

Министерство образования и науки РФ

```
алг Сумма квадратов (арг цел n, рез цел S)
  дано | n 0
  надо | S = 1*1 + 2*2 + 3*3 + ... + n*n
нач цел i
  ввод n; S:=0
  нц для i от 1 до n
   S:=S+i*i
  кц
  вывод "S = ", S
кон
```

программы

СКОЛКОВО
Московская школа управления

Министерство образования и науки РФ

```
void DressedDependOnWeather ( )
        {
            double Temperature; //Описываем новую переменную типа число с плавающей точкой
            Console.WriteLine("Введите температуру воздуха:"); //Вывод в консоль запроса
            Temperature = Convert.ToDouble(Console.ReadLine()); //Считывание с консоли введенного числа
            if(Temperature < 0.0) //Проверяем меньше ли 0 температура на улице
            {
              Console.WriteLine("Оденьте шубу"); //Вывод ответа в консоль
            }
            else
            {
              Console.WriteLine("Оденьте куртку"); //Вывод ответа в консоль
            }
```

СКОЛКОВО
Московская школа управления

Программа

Министерство образования и науки РФ

- ***Программа*** – формальная запись алгоритма на одном из языков программирования.
- Никлаус Вирт:

Программа = алгоритм + данные

- ***Компьютерная программа*** — последовательность инструкций, предназначенная для исполнения устройством управления вычислительной машины.

СКОЛКОВО
Московская школа управления

Определения

Министерство образования и науки РФ

- ***Программирование*** – процесс создания компьютерных программ.
- ***Программисты*** – люди занимающиеся программированием.
- ***Языки программирования*** - формальная знаковая система, предназначенная для записи компьютерных программ. Языки программирования могут быть реализованы как ***компилируемые*** и ***интерпретируемые***.

Классы языков программирования

- Языки низкого уровня:
  - Машинный код
  - Языки ассемблера
- Высокоуровневые языки:
  - Языки структурного программирования
  - Объектно-ориентированные языки
  - и.т.д.

СКОЛКОВО
Московская школа управления

Машинный код

Министерство образования и науки РФ

- Машинный код (native code) — система команд конкретной вычислительной машины (машинный язык), которая интерпретируется непосредственно микропроцессором или микропрограммами данной вычислительной машины.
- «Слова» машинного языка называются машинными инструкциями. Каждая из них описывает элементарное действие, выполняемое процессором, такое как «переслать байт из памяти в регистр». Программа — это просто длинный список инструкций, выполняемых процессором
- Программа «Hello, World!» для x86 (в шестнадцатеричном представлении побайтово):
- 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

СКОЛКОВО
Московская школа управления

Язык ассемблера

Министерство образования и науки РФ

- Язык ассемблера (автокод) — язык программирования низкого уровня. В отличие от языка машинных кодов, позволяет использовать более удобные для человека мнемонические (символьные) обозначения команд. При этом для перевода программы с языка ассемблера в понимаемый процессором машинный код требуется специальная программа, называемая ассемблером.
- Пример части программы на языке ассемблер для MS DOS:

```
mov ah,9
mov dx, OFFSET Msg
int 21h
int 20h
```

Структурное программирование

СКОЛКОВО
Московская школа управления

Министерство образования и науки РФ

- Методология разработки программного обеспечения, в основе которой лежит представление программы в виде иерархической структуры блоков. Предложена в 70-х годах XX века Э. Дейкстрой, разработана и дополнена Н. Виртом. В соответствии с данной методологией:
  - Любая программа представляет собой структуру, построенную из трёх типов базовых конструкций: последовательно выполнение, ветвление, цикл.
  - Повторяющиеся фрагменты программы могут оформляться в виде т. н. подпрограмм (процедур

программирование (ООП)

СКОЛКОВО
Московская школа управления

Министерство образования и науки РФ

- Объектно ориентированные языки реализуют концепцию ООП. ***ООП*** — парадигма программирования, основанная на представлении предметной области в виде системы взаимосвязанных абстрактных объектов и их реализаций.
- Основной проблемой процедурного программирования является то, что данные и функции их обработки не были связаны. С появлением концепции ООП появилась новая структура данных - ***Класс***. Это по сути дела тип данных, в котором кроме данных ***(свойств)*** также содержались функции их обработки ***(методы)***

СКОЛКОВО
Московская школа управления

Свойства ООП

Министерство образования и науки РФ

- ***Наследование***. Наследованием называется возможность порождать один класс от другого с сохранением всех свойств и методов класса-предка. Наследование призвано отобразить такое свойство реального мира, как иерархичность.
- ***Полиморфизм***. Полиморфизмом называют явление, при котором все классы-потомки имеют в наличии однотипные методы. Такие методы совпадают по форме (название, параметры, возвращаемые значения), но отличаются по реализации. Это позволяет обрабатывать объекты классов-потомков как однотипные объекты.
- ***Инкапсуляция***. Инкапсуляция — это принцип, согласно которому внутри класса можно совмещать данные (свойства) и описание действий (методы), через которые можно обращаться к свойствам.

СКОЛКОВО
Московская школа управления

Министерство образования и науки РФ

СЛОЖНОСТЬ АЛГОРИТМОВ

Анализ трудоёмкости алгоритмов

Целью анализа трудоёмкости алгоритмов является нахождение оптимального алгоритма для решения данной задачи.

***Теория сложности**, являясь частью **теории вычислений**, изучает ресурсы или стоимость вычислений, необходимые для выполнения поставленной проблемы.*

***Вычислительная сложность алгоритма** — это функция, определяющая зависимость объёма работы, выполняемой некоторым алгоритмом, от свойств входных данных. Объём работы обычно измеряется абстрактными понятиями времени и пространства, называемыми вычислительными ресурсами.*

***Время** определяется количеством элементарных шагов, необходимых для решения проблемы, тогда как **пространство** определяется объёмом памяти или места на носителе данных.*

Центральный вопрос разработки алгоритмов: «как изменится время исполнения и объём занятой памяти в зависимости от размера входа и выхода?».

Градации сложности

$f(n) = O(1)$ – *константная;*

$f(n) = O(\log n)$ – *логарифмическая; (см. пример ^ алгоритм 2)*

$f(n) = O(n)$ – *линейная; (см. пример * алгоритм 2)*

$f(n) = O(n^c)$ - *полиномиальная (квадратичная, кубическая ...); (см. пример* алгоритм 1)*

$f(n) = O(c^n)$ – *экспоненциальная; (см. пример ~ алгоритм 1)*

$f(n) = O(n!)$ – *факториальная*

*Такая **О оценка** дает нам верхнюю оценку временной трудоемкости алгоритма – его асимптотическую сложность.*

Примеры алгоритмов и их сложности

СКОЛКОВО
Московская школа управления

Министерство образования и науки РФ

Задача 1. *Вычислить n-ое число Фибоначчи (**полиномиальная сложность**):*

$$f_1 = 1, f_2 = 1, f_k = f_{k-2} + f_{k-1}$$

Алгоритм 1.1

- *Считаем рекурсивно «с конца» ($f_n, f_{n-1}, \ldots$);*

$$f_n = f_{n-2} + f_{n-1} \rightarrow f_{n-1} = f_{n-2} - f_{n-3} \rightarrow \cdots f_3 = f_1 + f_2$$

- *Сложность выполнения алгоритма $O(2^n)$*

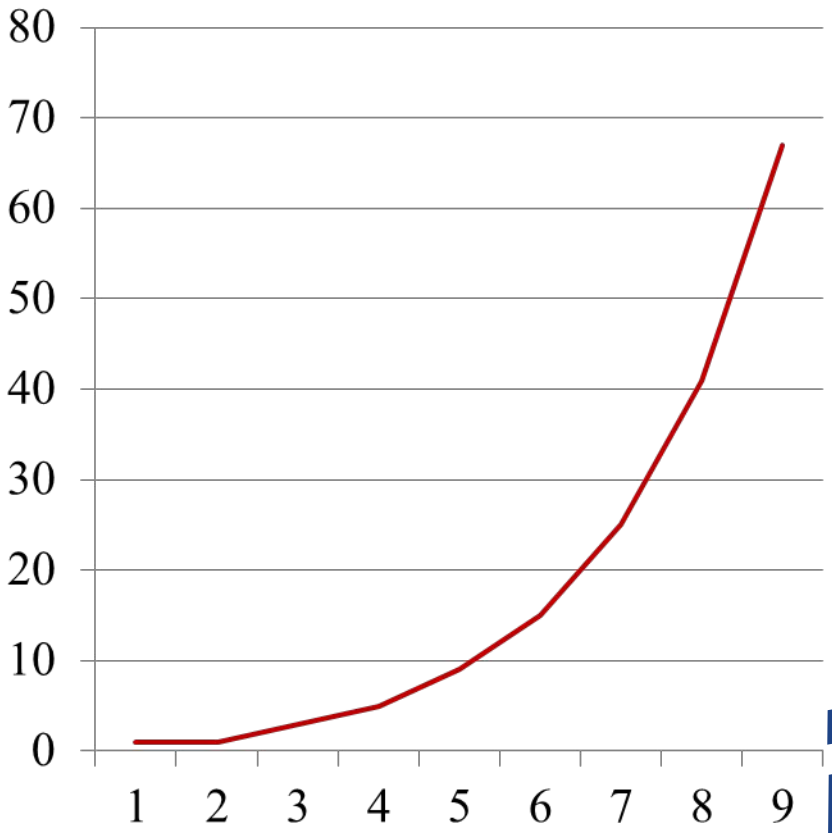
```
double FibFunc(int IndexNumber)//Передаем в функцию номер числа
{
   if (IndexNumber == 1 || IndexNumber == 2) //Проверяем условие дошли ли до
            1 или 2 числа
             return 1.0;//Если дошли, то возвращаем известные значения
   else
             return (FibFunc(IndexNumber - 1) + FibFunc(IndexNumber - 2));//Если
           нет, то рекурсивно вызываем сами себя используя формулу
}
```

Количество вызовов функции

СКОЛКОВО
Московская школа управления

Министерство образования и науки РФ

Номер числа	Число Фибоначчи	Число вызовов функций
1	1	1
2	1	1
3	2	3
4	3	5
5	5	9
6	8	15
7	13	25
8	21	41
9	34	67



Алгоритм 1.2

- *Считаем в цикле «с начала»* $(f_1, f_2, \ldots)$;

$$f_3 = f_1 + f_2 \rightarrow f_4 = f_2 + f_3 \rightarrow \cdots f_n = f_{n-2} + f_{n-1}$$

- *Сложность выполнения алгоритма* $O(n)$ ***(линейная)***

```
double FibFunc(int IndexNumber)
    {
      double f1, f2, f3; //Объявляем переменные
      f1 = f2 = 1.0; //Задаем извесные значения для 1го и 2го числа
      f3 = 0.0;
      int i = 3; //Объявляем и задаем значение переменной шага
      while (i <= IndexNumber)//Пока не достигнем необходимого номера числа
                                Фибоначчи
      {
        f3 = f2 + f1; // Определяем iое число по формуле
        //Меняем для следующего шага переменные
        f1 = f2; //f2 становится f1
        f2 = f3; //f3 становится f2
        i = i  + 1; //Увеличиваем шаг (номер числа) на единицу
      }
      return  f3; //Возвращаем последнее посчитанное значение
    }
```

СКОЛКОВО
Московская школа управления

Министерство образования и науки РФ

Задача 2. *Вычислить* a^n*:*

Алгоритм 2.1 (линейная сложность)

- *Считаем в цикле* $a, a^2, a^3, \ldots;$
- *Сложность выполнения алгоритма* $O(n)$

Алгоритм 1.2 (логарифмическая сложность)

- *Считаем используя свойство* $a^{2k} = (a^2)^k$
- *Сложность выполнения алгоритма* $O(log_2 n)$

```
long mypow(long a, long k)
        { long b = 1;
          while (k !=0 )
          {
            if (k % 2 == 0) {k /= 2;a *= a;}
                 else {k--; b *= a;}
          }
          return b;
        }
```

Количество итераций

Министерство образования и науки РФ

Степень	Число итераций
1	1
2	2
3	3
4	3
5	4
6	4
7	5
8	4
9	5
10	5
11	6
12	5
13	6
14	6