# Введение

Основные понятия и концепции

# Для чего полезно знать программирование?

1. Развитие алгоритмического мышления:
   - чтение и исполнение алгоритмов;
   - составление алгоритмов и документации;
   - восстановление алгоритмов и документации;
2. Расширение функционала имеющихся программ.
3. Понимание моделей естественных языков.
4. Понимание моделей искусственного интеллекта.

# Задачи, моделируемые методами искусственного интеллекта:

- распознавание образов (графической, графической текстовой, звуковой и т. п. информации);
- полуавтоматический перевод текста, написанного на естественных языках;
- определение местонахождения, идентификации, навигации и поиска;
- анализ данных;
- и т. д. и т. п.

# Основные понятия

- Что такое программирование?

Программирование — решение задач при помощи компьютеров.

# Области применения программирования:

- решение математических задач;
- системное программирование;
- прикладное программирование;
- скриптовое программирование;
- сетевое программирование;
- программирование под Web.

# Этапы разработки программ

1. **Постановка задачи** — определение требований к программному продукту.
2. **Анализ** — формализация постановки задачи и определение методов ее решения.
3. **Проектирование** — выбор структуры программного продукта, структуры данных, построение и оценка алгоритмов подпрограмм и определение особенностей взаимодействия программы с внешней средой (другими программами, ОС и техническими средствами).
4. **Реализация** — составление программы на выбранном языке программирования, ее тестирование и отладка.
5. **Модификация** — выпуск новых версий программного продукта.

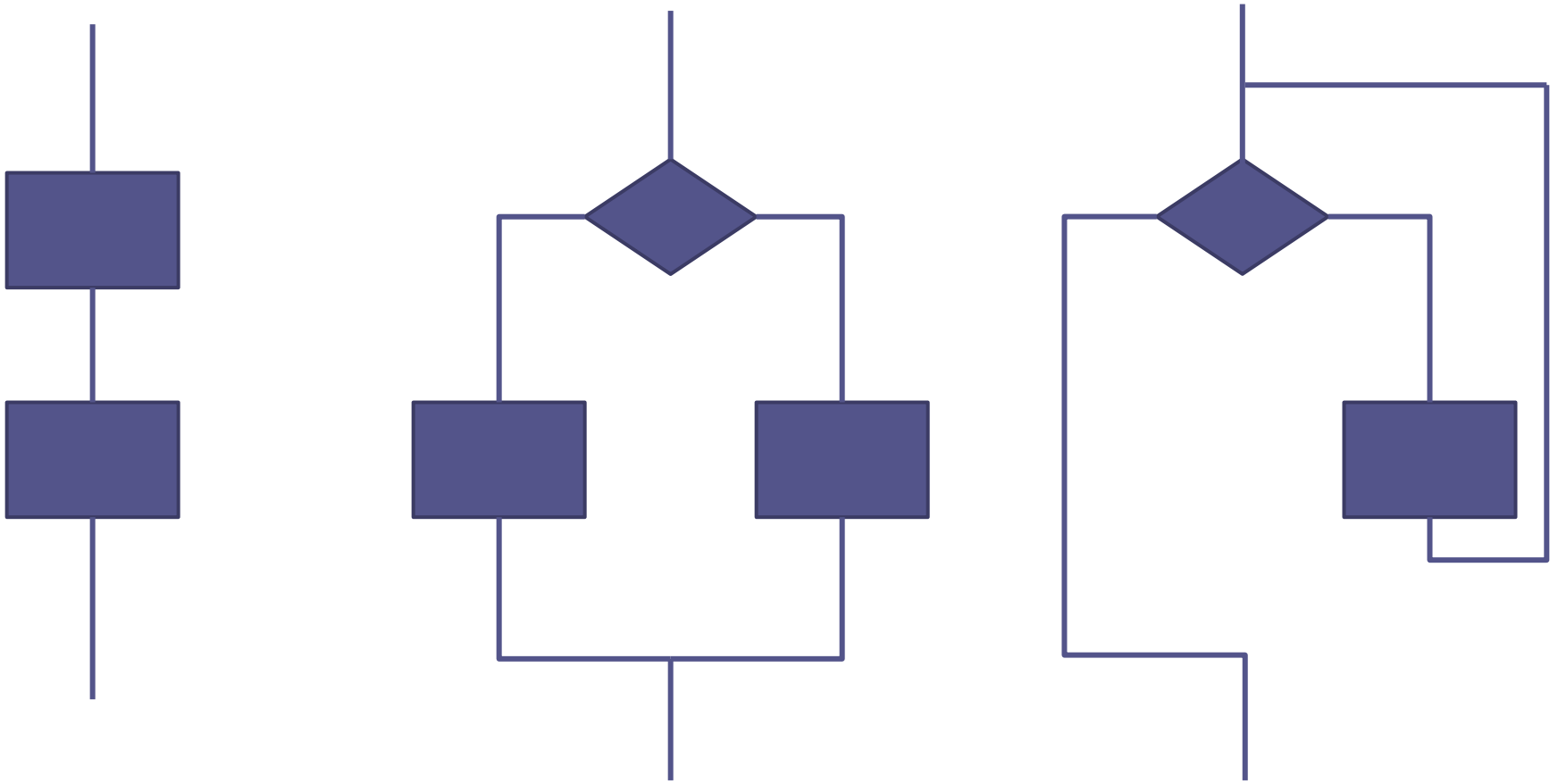
# Алгоритм

**Алгоритм** — формально описанная последовательность действий, которую надо выполнить для получения требуемого результата.

## Структуры последовательности действий

- линейная
- разветвленная
- циклическая

# Блок-схемы



# Псевдокод

Компактный (зачастую неформальный) язык описания алгоритмов, использующий ключевые слова и основные структуры языков программирования, но опускающий несущественные подробности и специфический синтаксис.

Псевдокод занимает промежуточное положение между естественным языком и формальным алгоритмическим языком.

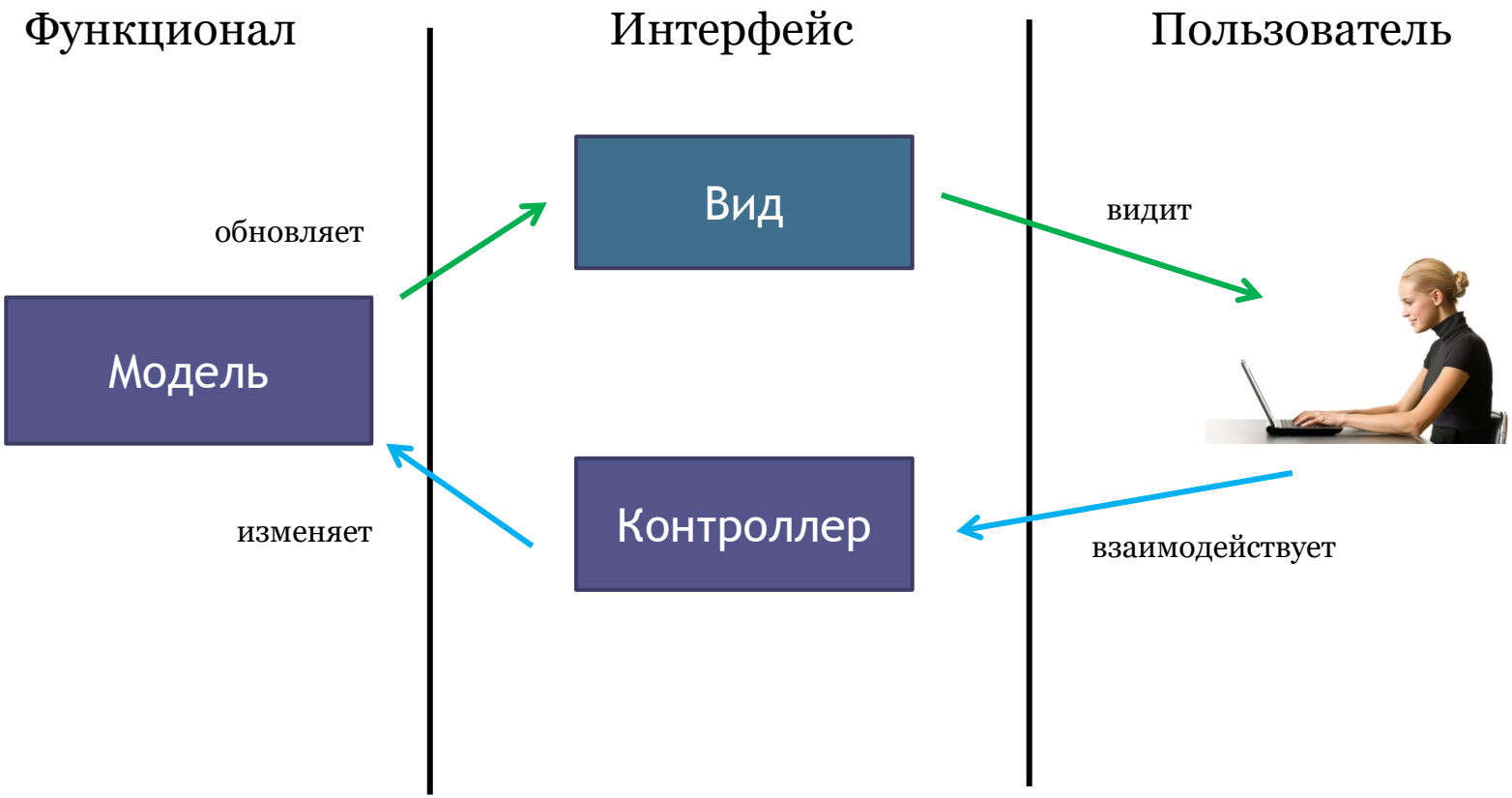
# Программа и ее составляющие

Программа — решение поставленной задачи.

Составляющие:

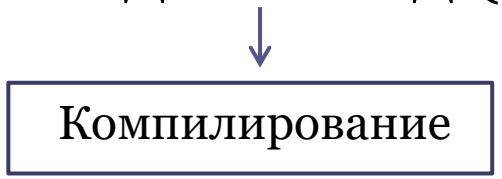
1. Функциональная составляющая.
2. Интерфейс.
3. Архитектура.

# Концепция MVC (model-viewer-controller)



# Эволюция кода

- Исходный код (текст) программы




- Объектный код

- Машинный код

# Среда программирования

- Текстовый редактор
- Компилятор
- Система отладки и поиска ошибок
- Справочная система

# Основные понятия

- **Операции** (встроенные) — сравнения, арифметические, логические, побитовые и т. д.
- **Операнд** — объект, над которым производится операция.
- **Идентификатор (имя)** — последовательность букв, цифр и специальных символов, начинающаяся с буквы или специального символа.
- **Переменная** — специальный объект, в котором хранятся данные. По сути это именованное место памяти.
- **Константа** — переменная, изменять значение которой в программе запрещено.

# Методология программирования

**Методология программирования** — это совокупность идей, понятий, принципов, способов и средств, определяющих стиль написания, отладки и сопровождения программ.

# Императивное программирование

1. **Процедурное программирование**. Это методология разработки ПО, в основе которой лежит процесс вычисления, описанный в виде инструкций.
2. **Модульное программирование** — методология программирования, при которой вся программа разбивается на группу компонентов, называемых модулями, причем каждый из них имеет свой контролируемый размер, четкое назначение и детально проработанный интерфейс с внешней средой.
3. **Структурное программирование** — методология разработки ПО, в основе которой лежит представление программы в виде иерархической структуры блоков.
4. **Объектно-ориентированное программирование (ООП)** — методология, основой которой являются классы и объекты.

# Структурное программирование

**Методология структурного императивного программирования** — подход, заключающийся в задании хорошей топологии императивных программ, в том числе отказе от использования глобальных данных и оператора безусловного перехода, разработке модулей с сильной связностью и обеспечении их независимости от других модулей.

**Три основных метода**, лежащих в основе структурного программирования:

1. **Метод модульной организации** частей программы заключается в разбиении программы на специальные компоненты, называемые модулями.
2. **Метод структурного кодирования** заключается в использовании при кодировании трех основных управляющих конструкций: последовательного исполнения, ветвления и цикла.
3. **Метод алгоритмической декомпозиции сверху вниз** заключается в пошаговой детализации алгоритма решения задачи от главной программы к подпрограммам самого нижнего уровня.

# Примеры программ на BASIC

Печать чисел от 1 до 10 и их квадратов.

**Неструктурированный код**

```
10    i = 0
20    i = i + 1
30    IF i <= 10 THEN GOTO 60
40    PRINT "Программа завершена."
50    END
60    PRINT i; " квадрат = "; i * i
70    GOTO 20
```

**Структурированный код**

```
FOR i = 1 TO 10
        PRINT i; " квадрат = "; i * i
NEXT i
PRINT "Программа завершена."
```

# Объектно-ориентированное программирование

В объектно-ориентированном программировании базовыми единицами программ и данных являются объекты.

**Объект —** это осязаемая сущность, которая четко проявляет свое поведение.

Объекты с одинаковыми свойствами и поведением абстрагируются в классы.

## Основные концепции ООП:

1. **Инкапсуляция** — возможность объединить данные и методы, работающие с ними в классе, и скрыть детали реализации от пользователя.
2. **Наследование** — возможность описать новый класс на основе уже существующего с частично или полностью заимствованной функциональностью.
3. **Полиморфизм** — возможность использовать объекты с одинаковым интерфейсом без информации о типе и внутренней структуре объекта.

# Декларативное программирование

Этот термин употребляется в двух смыслах. Первый — декларативная программа описывает **каково нечто, а не как его создать**. Второй — программа написана в соответствии с одной из трех следующих методологий:

1. **Логическое программирование** основано на автоматическом доказательстве теорем и принципах логического вывода информации на основе заданных фактов и правил вывода.
2. **Функциональное программирование —** методология, трактующая процесс вычислений как вычисление значений функций, понимаемых в математическом смысле, а не в смысле процедурного программирования (как подпрограмм).
3. **Программирование в ограничениях —** методология, в которой отношения между переменными указываются в форме ограничений.

# Быстрая сортировка (Pascal)

```
procedure qSort(var ar: array of real);
    procedure sort(var ar: array of real; low, high: integer);
    var i, j: integer;
        m, wsp: real;
        begin
            i:=low; j:=high; m:=ar[(i+j) div 2];
            repeat
                while ar[i]<m do Inc(i);
                while ar[j]>m do Dec(j);
                if i<=j then begin
                        wsp:=ar[i];  ar[i]:=ar[j];
                        ar[j]:=wsp; Inc(i);
                        Dec(j);
                end;
            until i>j;
            if low<j then sort(ar, low, j-1);
            if i<high then sort(ar, i+1, high);
        end;
    begin
        sort(ar, 0, High(ar));
    end;
```

# Быстрая сортировка (Haskell)

```
qsort([]) -> []
qsort([h|t]) ->
    qsort([ x | x <- t, x < h]) ++ [h] ++
    qsort([ y | y<- t, y >= h])
```