

ЛЕКЦИЯ 03

АЛГОРИТМ. ЭТАПЫ РЕШЕНИЯ ЗАДАЧ.

ОСНОВЫ АЛГОРИТМИЗАЦИИ И
ПРОГРАММИРОВАНИЯ



ОСНОВНЫЕ ПОНЯТИЯ

Задача — проблемная ситуация с явно заданной целью, которую необходимо достичь.

Алгоритм — это последовательность команд управления каким-либо исполнителем

ОСНОВНЫЕ ЭТАПЫ

1. Постановка задачи.
2. Формализация задачи.
3. Построение алгоритма.
4. Составление программы на языке программирования.
5. Отладка и тестирование программы.
6. Проведение расчётов и анализ полученных результатов

ПОСТАНОВКА ЗАДАЧИ

На данном этапе формулируется цель решения задачи, описывается её содержание. Должны быть чётко определены исходные данные, необходимые для решения задачи, и результат.

Например

Необходимо рассчитать полную стоимость обучения в колледже/ВУЗе, если известны стоимость обучения за 1 год и длительность обучения.

ПОСТАНОВКА ЗАДАЧИ. ПРИМЕР

Определим исходные данные и требуемый результат

ПОСТАНОВКА ЗАДАЧИ. ПРИМЕР

Определим исходные данные и требуемый результат

Исходные данные:

- стоимость обучения за 1 год, руб (pr)
- длительность обучения, лет (a)

ПОСТАНОВКА ЗАДАЧИ. ПРИМЕР

Определим исходные данные и требуемый результат

Исходные данные:

- стоимость обучения за 1 год, руб (pr)
- длительность обучения, лет (a)

Результат:

- Полная стоимость обучения, руб (Z)

ФОРМАЛИЗАЦИЯ ЗАДАЧИ

Компьютер "умеет работать только с числами". Все действия центрального процессора - это арифметические операции над числовыми данными. Поэтому, при решении задачи с помощью компьютера необходимо перевести задачу на язык математики. Чаще всего, решение задачи сводится к математическому описанию какого-то объекта, явления или процесса. Другими словами, *этап формализации сводится к получению математической модели.*

ФОРМАЛИЗАЦИЯ ЗАДАЧИ. ПРИМЕР

Необходимо определить, как связаны стоимость 1 года обучения и длительность обучения в полной стоимости с точки зрения математики.

ФОРМАЛИЗАЦИЯ ЗАДАЧИ. ПРИМЕР

Необходимо определить, как связаны стоимость 1 года обучения и длительность обучения в полной стоимости с точки зрения математики.

Получим следующую зависимость:

$$Z = pr \cdot a$$

ПОСТРОЕНИЕ АЛГОРИТМА

Алгоритм - это детерминированная последовательность команд управления каким-либо исполнителем, которая приводит к заведомо известному результату

СВОЙСТВА АЛГОРИТМА

1. Массовость (универсальность).
2. Дискретность (разрывность).
3. Определенность (точность, детерминированность)
4. Понятность.
5. Формальность.
6. Завершаемость (конечность).
7. Результативность.

СВОЙСТВА АЛГОРИТМА

1. **Массовость** (универсальность). Благодаря этому свойству, алгоритм можно успешно применять к различным наборам исходных данных. Пусть существует запись некой абстрактной последовательности, выраженная формулой. Подставляя в эту формулу значения (каждый раз новые), пользователь будет получать верные решения в соответствии с определенным алгоритмом действий.

СВОЙСТВА АЛГОРИТМА

2. **Дискретность** (разрывность). Это свойство характеризует структуру. Каждая алгоритмическая последовательность действий делится на этапы (шаги), а процесс решения задачи — это последовательное исполнение простых шагов. Также дискретность обозначает, что для выполнения каждого этапа потребуется конечный временной отрезок (исходные данные преобразуются во времени в результат дискретно).

СВОЙСТВА АЛГОРИТМА

3. **Определенность** (точность, детерминированность) — это свойство указывает алгоритму, что каждый его шаг должен быть строго определенным, то есть различные толкования должны быть исключены. В результате каждый шаг определяется состоянием системы однозначно, когда четко понятно, какая команда станет выполняться на следующем шаге. Как итог — при любом исполнителе для одних и тех же исходных данных при выполнении одной и той же цепочки команд будет выдаваться одинаковый результат.

А что насчет генератора случайных чисел? Подумайте над этим

СВОЙСТВА АЛГОРИТМА

4. **Понятность.** Должны быть включены лишь те команды, которые доступны и понятны исполнителю, то есть входят в систему его команд.

СВОЙСТВА АЛГОРИТМА

5. **Формальность.** Любой исполнитель действует формально и лишь выполняет инструкции, не вникая в смысл. Он не отвлекается от поставленной задачи и не рассуждает, зачем и почему они нужны. Рассуждениями занимается разработчик алгоритма, задача же исполнителя — просто исполнить предложенные команды и получить результат. «Приказы не обсуждают, а выполняют».

СВОЙСТВА АЛГОРИТМА

6. **Завершаемость** (конечность). Если исходные данные заданы корректно, алгоритм завершит свое действие и выдаст результат за конечное число шагов.

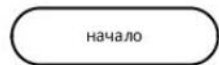
СВОЙСТВА АЛГОРИТМА

7. **Результативность.** Согласно этому свойству, любой алгоритм должен завершаться конкретными результатами.

СПОСОБЫ ОПИСАНИЯ АЛГОРИТМА

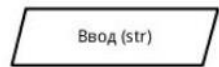
1. Словесный (описание учителем порядка решения квадратных уравнений в школе)
2. Текстовый (рецепт приготовления блюда)
3. Графический (блок-схема)

БЛОК-СХЕМА. ОСНОВНЫЕ ОБОЗНАЧЕНИЯ



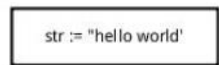
Терминатор начала и конца
работы функции

Терминатором начинается и заканчивается любая функция. Тип возвращаемого значения и аргументов функции обычно указывается в комментариях к блоку терминатора.



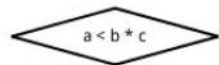
Операции ввода и вывода
данных

В ГОСТ определено множество символов ввода/вывода, например вывод на магнитные ленты, дисплеи и т.п. Если источник данных не принципиален, обычно используется символ параллелограмма. Подробности ввода/вывода могут быть указаны в комментариях.



Выполнение операций над
данными

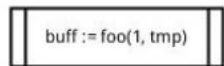
В блоке операций обычно размещают одно или несколько (ГОСТ не запрещает) операций присваивания, не требующих вызова внешних функций.



Блок, иллюстрирующий
ветвление алгоритма

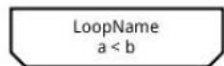
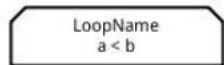
Блок в виде ромба имеет один вход и несколько подписанных выходов. В случае, если блок имеет 2 выхода (соответствует оператору ветвления), на них подписывается результат сравнения — «да/нет». Если из блока выходит большее число линий (оператор выбора), внутри него записывается имя переменной, а на выходящих дугах — значения этой переменной.

БЛОК-СХЕМА. ОСНОВНЫЕ ОБОЗНАЧЕНИЯ



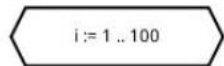
Вызов внешней процедуры

Вызов внешних процедур и функций помещается в прямоугольник с дополнительными вертикальными линиями.



Начало и конец цикла

Символы начала и конца цикла содержат имя и условие. Условие может отсутствовать в одном из символов пары. Расположение условия, определяет тип оператора, соответствующего символам на языке высокого уровня — оператор с предусловием (while) или постусловием (do ... while).



Подготовка данных

Символ «подготовка данных» в произвольной форме (в ГОСТ нет ни пояснений, ни примеров), задает входные значения. Используется обычно для задания циклов со счетчиком.



Соединитель

В случае, если блок-схема не умещается на лист, используется символ соединителя, отражающий переход потока управления между листами. Символ может использоваться и на одном листе, если по каким-либо причинам тянуть линию не удобно.

БЛОК-СХЕМА. ОСНОВНЫЕ ОБОЗНАЧЕНИЯ

Внутри блоков блок-схемы указываются математические выражения, словесное описание алгоритма на естественном языке или *псевдокод*.

Псевдокод - компактный, зачастую неформальный язык описания алгоритмов, использующий ключевые слова императивных языков программирования, но опускающий несущественные для понимания алгоритма подробности и специфический синтаксис.

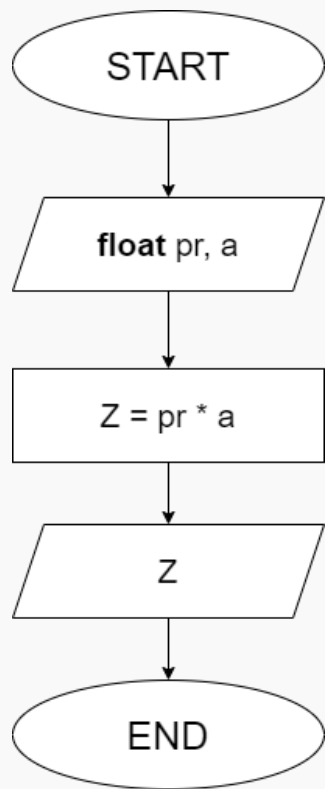
Алгоритм должен быть описан таким образом, чтобы программист, владеющий **любым** языком программирования мог его понять и реализовать.

ПОСТРОЕНИЕ АЛГОРИТМА

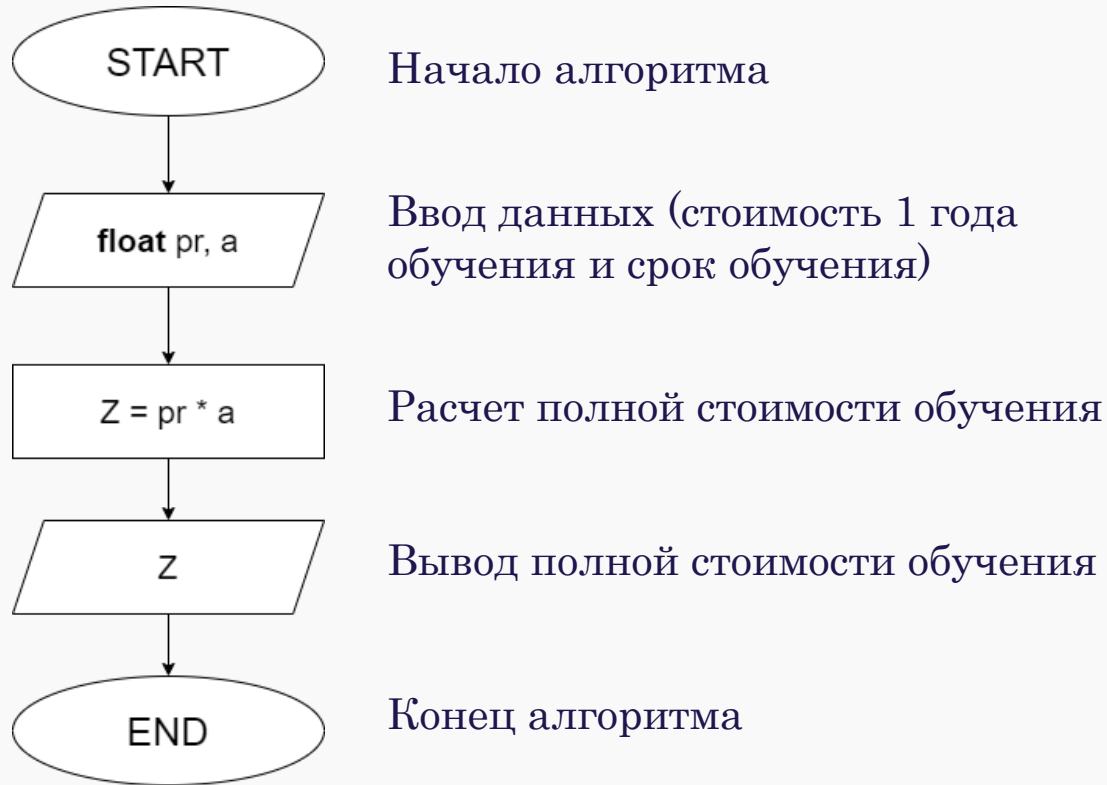
В очень простых задачах (например, как наша) данный этап можно пропустить, и сразу писать код программы на языке программирования. Модель настолько простая, что последовательность действий здесь видна явно, и описывать алгоритм (используя блок-схемы или псевдокод) нет необходимости. Но в обучающих целях построим алгоритм для нашей задачи.

В наших задачах исполнителем будет выступать компьютер, который работает с числами, хранящимися в памяти - величинами.

ПОСТРОЕНИЕ АЛГОРИТМА



ПОСТРОЕНИЕ АЛГОРИТМА



СОСТАВЛЕНИЕ ПРОГРАММЫ НА ЯЗЫКЕ ПРОГРАММИРОВАНИЯ

Для выполнения первых трёх этапов, как видим, компьютер не нужен. Задача решается не на компьютере, *задача решается в голове путём логических рассуждений*. Получив математическую модель - мы, по сути, уже решили задачу.

Компьютер - это инструмент, который позволяет ускорить данное решение задачи применимо уже к конкретным исходным данным.

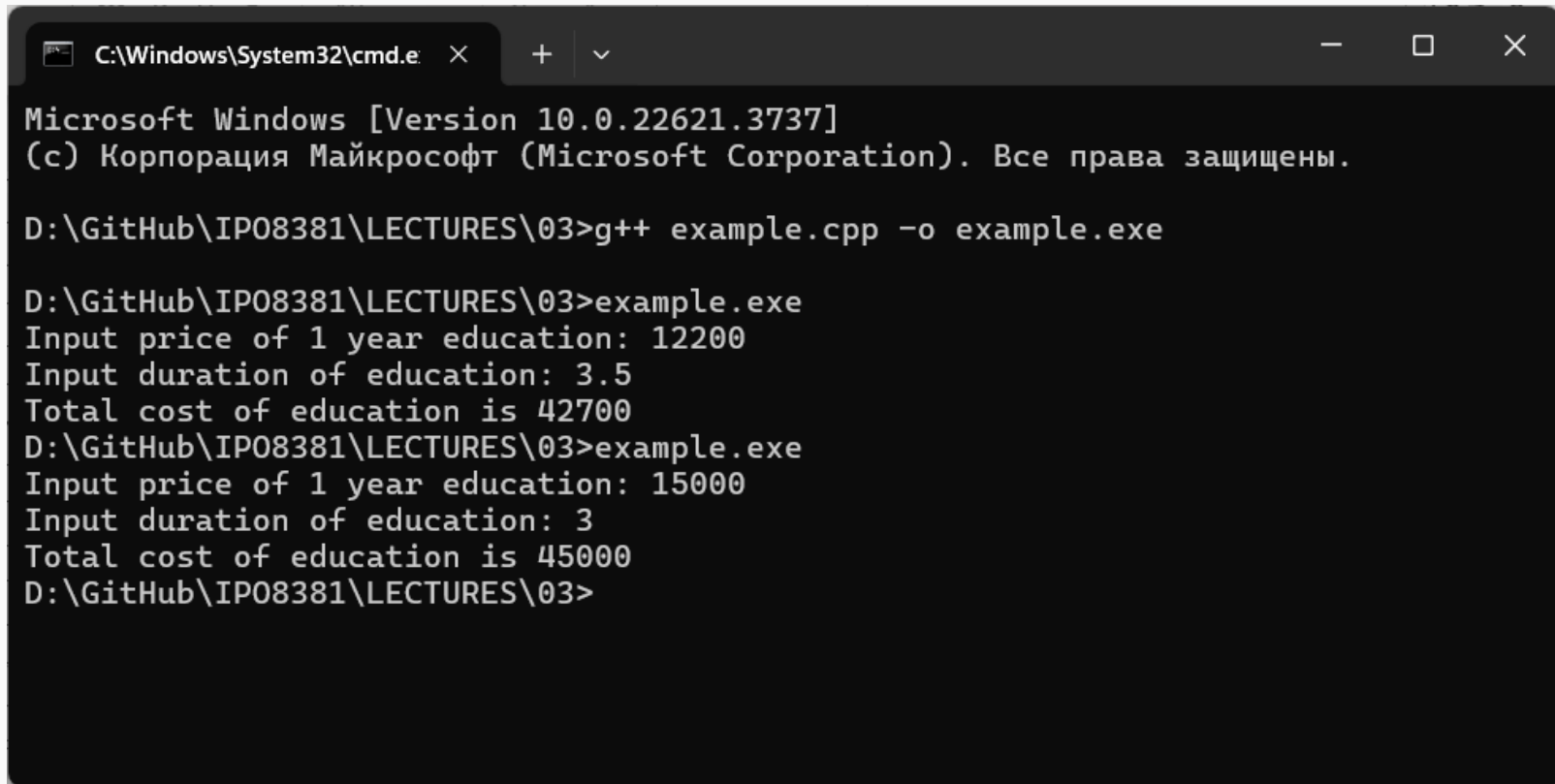
Имея модель задачи и готовый алгоритм, можно переходить к этапу написания кода программы

СОСТАВЛЕНИЕ ПРОГРАММЫ НА ЯЗЫКЕ ПРОГРАММИРОВАНИЯ

```
#include <iostream>

int main() {
    float pr = 0; //Переменная для стоимости 1 года обучения
    float a = 0; //Переменная для длительности обучения
    std::cout << "Input price of 1 year education: "; //Приглашение пользователю
    ввести данные
    std::cin >> pr; //Чтение полученных данных
    std::cout << "Input duration of education: "; //Приглашение пользователю
    ввести данные
    std::cin >> a; //Чтение полученных данных
    float Z = pr * a; //Вычисление полной стоимости обучения
    std::cout << "Total cost of education is " << Z; //Вывод полной стоимости
    обучения
    return 0; //Завершение работы программы
}
```

РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ



```
C:\Windows\System32\cmd.e  X  +  v  -  □  X

Microsoft Windows [Version 10.0.22621.3737]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

D:\GitHub\IP08381\LECTURES\03>g++ example.cpp -o example.exe

D:\GitHub\IP08381\LECTURES\03>example.exe
Input price of 1 year education: 12200
Input duration of education: 3.5
Total cost of education is 42700
D:\GitHub\IP08381\LECTURES\03>example.exe
Input price of 1 year education: 15000
Input duration of education: 3
Total cost of education is 45000
D:\GitHub\IP08381\LECTURES\03>
```

ОТЛАДКА И ТЕСТИРОВАНИЕ ПРОГРАММЫ

Этот этап считается наиболее трудоёмким. Цель тестирования программы - это выявление ошибок, которые могли возникнуть на этапе формализации и разработки алгоритма, а также при написании самого кода программы.

ОТЛАДКА И ТЕСТИРОВАНИЕ ПРОГРАММЫ

На этапе тестирования выявляют следующие виды ошибок:

Синтаксические ошибки — это ошибки в записи конструкций языка программирования. Обнаружение большинства синтаксических ошибок автоматизировано в основных системах программирования. Они выявляются уже на этапе написания программы.

Семантические ошибки — это ошибки, связанные с неправильным содержанием действий и использованием недопустимых значений величин.

Логические ошибки - это ошибки в самом алгоритме. Данные ошибки формируются на этапе разработки алгоритма при неправильном решении задачи. Выявление такой ошибки зачастую требует повторного решения задачи с самого первого этапа.

ОТЛАДКА И ТЕСТИРОВАНИЕ ПРОГРАММЫ

На этапе тестирования выявляют следующие виды ошибок:

Синтаксические ошибки — это ошибки в записи конструкций языка программирования. Обнаружение большинства синтаксических ошибок автоматизировано в основных системах программирования. Они выявляются уже на этапе написания программы.

Семантические ошибки — это ошибки, связанные с неправильным содержанием действий и использованием недопустимых значений величин.

Логические ошибки - это ошибки в самом алгоритме. Данные ошибки формируются на этапе разработки алгоритма при неправильном решении задачи. Выявление такой ошибки зачастую требует повторного решения задачи с самого первого этапа.

ОТЛАДКА И ТЕСТИРОВАНИЕ ПРОГРАММЫ

На этапе тестирования выявляют следующие виды ошибок:

Синтаксические ошибки — это ошибки в записи конструкций языка программирования. Обнаружение большинства синтаксических ошибок автоматизировано в основных системах программирования. Они выявляются уже на этапе написания программы.

Семантические ошибки — это ошибки, связанные с неправильным содержанием действий и использованием недопустимых значений величин.

Логические ошибки - это ошибки в самом алгоритме. Данные ошибки формируются на этапе разработки алгоритма при неправильном решении задачи. Выявление такой ошибки зачастую требует повторного решения задачи с самого первого этапа.

ПРОВЕДЕНИЕ РАСЧЁТОВ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ.

Этот этап подразумевает использование готовой программы в практических целях для решения поставленной практической задачи. В нашем примере мы можем использовать нашу программу для вычисления стоимости обучения в зависимости от тарифов на обучение в учебном заведении и длительности обучения

ДОМАШНЕЕ ЗАДАНИЕ

1. Решить задачу расчета стоимости краски, которая необходима для покраски всех стен в помещении, если известны параметры помещения и стоимость краски. (Выполнить этапы 1 - 5)
2. Решить задачу нахождения корня уравнения вида $kx + b = a$, если известны параметры k , b и a (Выполнить этапы 1 - 5)
3. Решить задачу нахождения корней квадратного уравнения. (Выполнить этапы 1 - 5)
4. Выучить этапы решения задач на компьютере

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

1. Дорохова Т.Ю., Основы алгоритмизации и программирования : учебное пособие для СПО / Т.Ю. Дорохова, И.Е. Ильина. — Саратов, Москва : Профобразование, Ай, Пи Ар Медиа, 2022. — 139 с.
2. Кудинов Ю.И., Основы алгоритмизации и программирования : учебное пособие для СПО / Ю.И. Кудинов, А.Ю. Келина. — 2-е изд. — Липецк, Саратов: Липецкий государственный технический университет, Профообразование, 2020. — 71 с.
3. Дональд Кнут, Искусство программирования. Том 1. Основные алгоритмы / Ю.В. Козаченко. - 3-е изд — Москва, Санкт-Петербург: ВИЛЬЯМС, 2018. — 721 с.