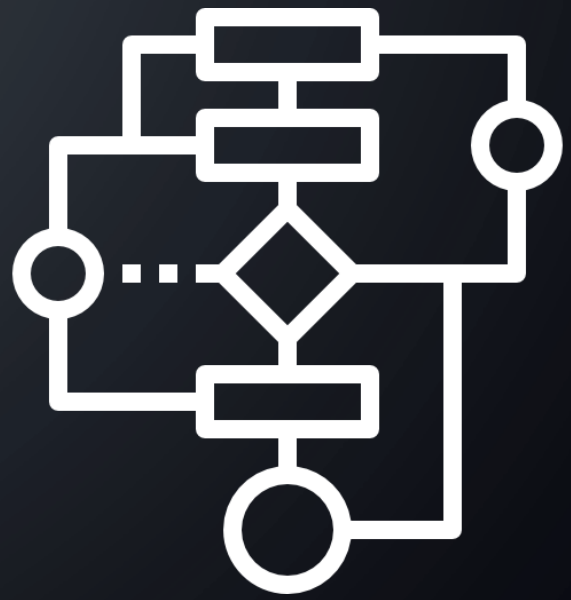
Лекция №3
по дисциплине
«ТЕОРИЯ АЛГОРИТМОВ»

ПОНЯТИЕ И ВИДЫ АЛГОРИТМОВ

Преподаватель:
Золотоверх Д.О.

АЛГОРИТМ

- Алгоритм – последовательность чётко определенных действий, выполнение которых ведёт к решению задачи.
- Алгоритм – это совокупность действий, приводящих к достижению результата за конечное число шагов.
- Часто в качестве исполнителя выступает компьютер, но понятие алгоритма необязательно относится к компьютерным программам.

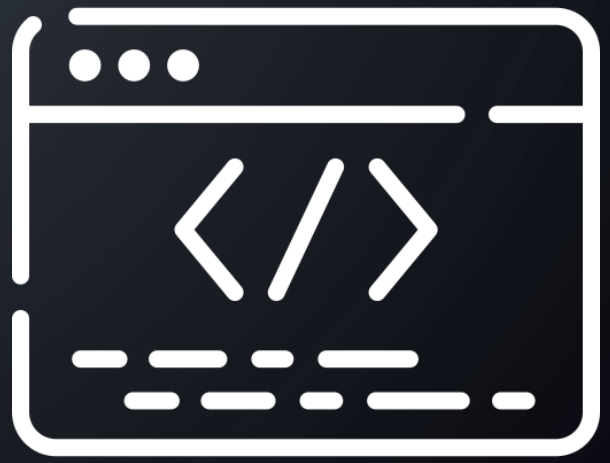


ПРОГРАММА

Программа – это реализация алгоритма для выполнения задачи компьютером.

С помощью программы мы формулируем алгоритм на языке, что понятен компьютеру.

Программа пишется с помощью языка программирования.



ЯЗЫК ПРОГРАММИРОВАНИЯ

Язык программирования — формальный язык, предназначенный для записи компьютерных программ.

Язык программирования определяет набор лексических, синтаксических и семантических правил.

Лексика – словарный состав языка;

Синтаксис - правила, с помощью которых пишется программа на уровне алфавита;

Семантика – правила, определяющие конструкции языка программирования.

ЯЗЫКИ ПРОГРАММИРОВАНИЯ

- Языков существует очень много
- Языки различаются
- Языки могут выполнять разные задачи
- В некоторых сферах одни языки являются более предпочтительней других
- Рейтинг языков TOIBE, PYPL

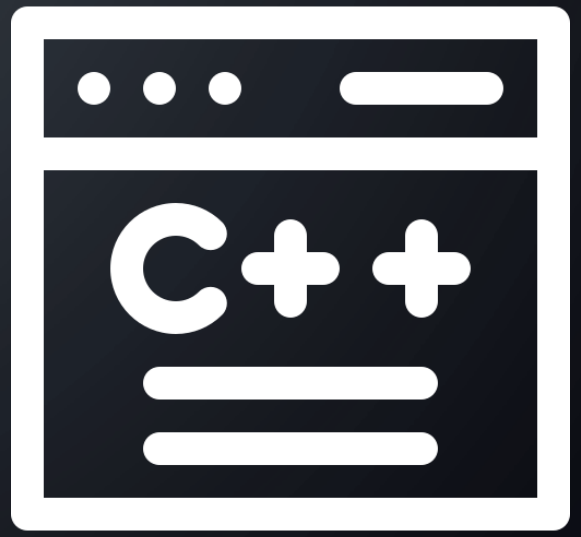
ЯЗЫК ПРОГРАММИРОВАНИЯ C++

Это компилируемый, статически типизированный язык программирования общего назначения.

Поддерживает множество парадигм программирования

Содержит свойства высоко-, а в некоторых случаях низкоуровневого языка.

Синтаксис C++ унаследован от языка C.

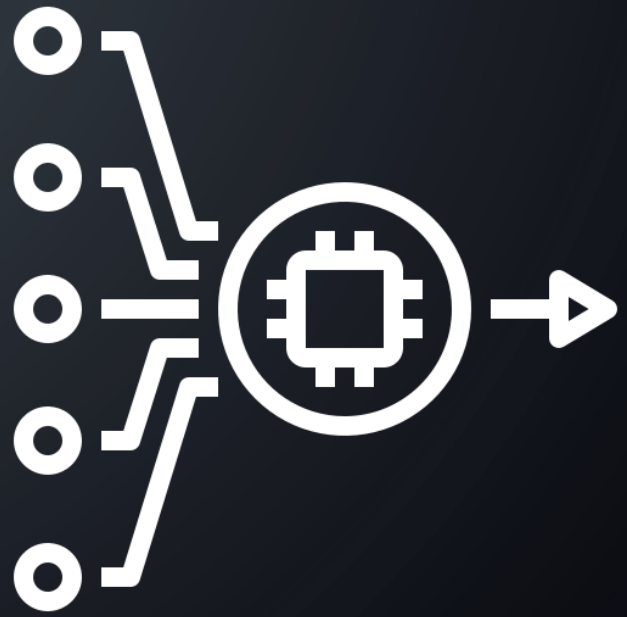


КОМПИЛИРУЕМЫЙ ЯЗЫК

Компилятор — программа, переводящая текст, написанный на языке программирования, в машинный код.

Производит сборку программы:

- трансляция модулей программы, написаных на высокоуровневом языке в низкоуровневый.
- вставка модулей из используемых библиотек
- генерация кода запроса к ОС



ПРИМЕР КОМПИЛЯЦИИ

ИСХОДНЫЙ КОД

```
#include <iostream>
using namespace std;

int main() {
    cout << "Hello, World!" << endl;
}
```

МАШИННЫЙ КОД

```
457f 464c 0102 0001 0000 0000 0000 0000
0003 003e 0001 0000 10c0 0000 0000 0000
0040 0000 0000 0000 3be8 0000 0000 0000
0000 0000 0040 0038 000d 0040 001f 001e
0006 0000 0004 0000 0040 0000 0000 0000
0040 0000 0000 0000 0040 0000 0000 0000
02d8 0000 0000 0000 02d8 0000 0000 0000
0008 0000 0000 0000 0003 0000 0004 0000
0318 0000 0000 0000 0318 0000 0000 0000
0318 0000 0000 0000 001c 0000 0000 0000
001c 0000 0000 0000 0001 0000 0000 0000
0001 0000 0004 0000 0000 0000 0000 0000
```

ИНТЕРПРЕТАТОР

Выполняет построчный анализ, обработку и выполнение исходного кода программы.

Ключевое отличие от компиляции – весь текст программы, перед запуском не анализируется и не транслируется в машинный код, и лишь ее часть.

КОМПИЛЯТОР VS ИНТЕРПРЕТАТОР

Компилируемая программа имеет след. преимущества:

- Может запускаться без посторонних программ
- Наличие различных оптимизаций
- Намного быстрее интерпретированной

Недостатки:

- Малая переместимость
- Менее совершенные и не наглядные средства диагностики ошибок
- Большие размеры программ

КОМПИЛЯТОР VS ИНТЕРПРЕТАТОР

C++

```cpp
#include <iostream>
using namespace std;

int main() {
    int quantity;
    int previous = 0;
    int next = 1;
    int buffer = next;
    cin >> quantity;
    while (quantity > 0) {
        cout << next << endl;
        buffer = next;
        next += previous;
        previous = buffer;
        quantity--;
    }
}
```

Python

```python
from fibonacci import fibonacci

print(fibonacci(length=10))
```

ТИПИЗАЦИЯ

Бывает динамическая и статическая:

- Статическая – типы данных значения переменной не может меняться при выполнении программы.

Как следствие – простой машинный код, но сложнее процесс программирование.

- Динамическая – наоборот, значения могут менять тип данных.

ПАРАДИГМА ПРОГРАММИРОВАНИЯ

это совокупность идей и понятий, определяющих стиль написания компьютерных программ.

Парадигма может не определяется языком (их допускается несколько).

Самые распространённые парадигмы для C++:

- процедурная;
- обобщенная;
- объектно-ориентированная.

«ВЫСОКОСТЬ» ЯЗЫКА

Высокоуровневый язык программирования разработанный для быстроты и удобства в использовании программистом.

Основная черта высокоуровневых языков — это абстракция, наличие конструкций, кратко описывающих такие структуры данных и операции, описания которых на машинном коде очень длинны и сложны для понимания.

ПРИМЕР ПРОГРАММЫ C++

`#include <название библиотеки>` - подключение библиотеки (ее заголовочного файла)

`using namespace std` – использования имен

`int main() {}` - главная функция программы, выполняется, когда запускается программа.

тело функции - записываются действия и операции, предусмотренные алгоритмом.

```
#include <iostream>
using namespace std;

int main() {
    int quantity;
    int previous = 0;
    int next = 1;
    int buffer = next;
    cin >> quantity;
    while (quantity > 0) {
        cout << next << endl;
        buffer = next;
        next += previous;
        previous = buffer;
        quantity--;
    }
}
```

СПАСИБО ЗА ВНИМАНИЕ!