Среды программирования

Программные средства (ПО, Soft)

Программирование -

это процесс создания программы для решения задачи с помощью ЭВМ

Основные этапы технологического процесса решения задач с помощью ЭВМ

1 этап: Постановка задачи и выбор метода решения (формальное математическое описание алгоритма)

2 этап: Определение и описание входных и выходных данных, необходимых для решения задач.

3 этап: Разработка алгоритма решения задач.

4 этап: Кодирование описания данных и алгоритма (составление программы на выбранном языке программирования).

5 этап: Отладка и тестирование программы с целью её проверки и доведения её в соответствии с поставленной задачей.

6 этап: Выполнение и поддержка программы (создание новых версий в зависимости от новой техники).

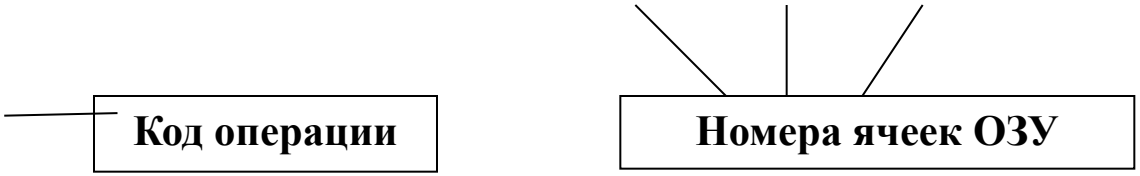
Понятие программы

1. **Программа** – это форма представления алгоритма для исполнения его компьютером
2. **Программа** – последовательность инструкций (кодов), предназначенная для исполнения устройством управления вычислительной машины

Первые команды составлялись в **машинных кодах.**

Например, **команда сложения** двух чисел могла выглядеть так:

15 0127 2677 3656



Понятнее записать команду так: **C = A + B**

Здесь латинские буквы **A ,B ,C** обозначают **переменные**

Для автоматизации формирования машинного кода нужно было решить 2 задачи:

1 задача

Создать систему условных обозначений для записи команд в понятной для человека форме (**язык программирования**)

2 задача

Создать **программу-посредника**, которая переводила бы такие команды на машинный язык.

Реализация 1-й задачи

- Алгоритм должен быть записан на алгоритмическом языке (АЯ), чтобы быть исполненным.
- Команды на языке программирования называют операторами или инструкциями
- Программа, написанная на языке программирования, – это последовательность операторов (или исходный текст)
- Исходные тексты программ хранятся в текстовых файлах
- Программа должна иметь и машинный код, который непосредственно исполняется

Реализация 2-й задачи

Существует **два типа программ-посредников**, работающих с исходными текстами:

1. **Компилятор:**

   - переводит исходный текст в машинный код,
   - записывает машинный код на диск в форме исполняемого (загрузочного) файла.

   После этого программа выполняется независимо от исходного текста.

2. **Интерпретатор:**

   - интерпретирует каждую инструкцию исходного текста и немедленно ее исполняет,
   - файл на машинном языке не создается.

Замечание. Программа в режиме Интерпретации работает медленнее, но проще для отладки

Схема работы компилятора

Общие особенности языков программирования

1. Это формализованные (искусственные) языки – в них строго определены **синтаксис** и **семантика**:
   - синтаксис описывает структуру программ как наборов символов (безотносительно к содержанию)
   - семантика определяет смысловое значение отдельных языковых конструкций
2. Все языки содержат:
   - средства описания данных,
   - арифметические операторы,
   - средства управления и организации циклов,
   - средства ввода и вывода информации.
3. Многие языки используют похожие принципы организации программ, но разный синтаксис.

Требования к языку программирования:

– программа должна быть пригодна для восприятия компьютером,

– программа должна быть понятной для человека.

Основные категории языков программирования

Языки высокого уровня (high-level language):

- наглядное описание задачи,
- не зависит от внутренних машинных кодов ЭВМ,
- требует наличие транслятора или интерпретатора.

Pascal, C, C++, C#, Java, …

Языки низкого уровня (low-level language):

- предназначен для определенного типа ЭВМ,
- отражает его внутренний машинный код.

Ассемблер, Макроассемблер

Основные подходы к программированию

Процедурное (процедурно-ориентированное) - в основу положен модульный (структурный) принцип:

Программа - это последовательность процедур или функций, т.е. последовательность действий.

Языки программирования: Фортран, Паскаль, Си

ООП (объектно-ориентированное) - в основу положена концепция объекта:

Объект = данные + выполняемые над ними действия (процедуры или функции).
Программа – это набор объектов и связей между ними.

Языки программирования: C++, Java, ObjectLisp

Структурное программирование:

1- каждая задача разбивается на какие-то цельные завершенные части (модули),
2- программирование ведется исключительно по этим частям - написали часть номер 1, протестировали ее, написали часть номер 2, протестировали ее…
3 - потом все вместе собрали и получили программный продукт.

```
{          Вариант 1: Вычисление площади круга          }
Program PRIM1;                                  {заголовок программы}
 Uses Crt;          {подключение модуля управления экраном в текстовом режиме}
Const                                           {подраздел объявления констант}
    Pi=3.14;                                    {задание константы Pi}
Var                                             {подраздел объявления переменных}
    R,S: Real;          {переменные вещественного типа – радиус и площадь круга}
Begin                                           {начало раздела операторов}
  Clrscr;                                       {очистка экрана}
  R:=1.5;                {оператор присваивания переменной R значения 1.5}
  S:=Pi*R*R;                                    {оператор присваивания для вычисления S }
  Writeln('S=',S:7:3);                          {вывод на экран значения S}
End.                                            {конец программы}
```

```
        /*   Вариант1: Вычисление площади круга */
#include <stdio.h>    //директива препроцессора для подключения стандартной библиотеки
                       // ввода - вывода
void main()                  // заголовок функции main()
{
  const float Pi=3.14;         // определение вещественной константы Pi равной 3.14
   float R,S;                  // оператор описания вещественных переменных R и S
   R=1.5;                      // оператор присваивания переменной R значения 1.5
   S=Pi*R*R;                   // оператор присваивания S вычисленного значения
   printf("S=%f\n",S);         // оператор вывода на экран значения S
}
```

```
{Вариант 1: Вычисление площади круга}
Program PRIM1;
Uses WinCrt;
Const
    Pi=3.14;
Var
    R,S: Real;
Begin
  Clrscr;
  R:=1.5;
  S:=Pi*R*R;
  Writeln('S=',S:7:3);
End.
```

```
/*   Вариант1: Вычисление площади круга */
#include <stdio.h>
void main()
{
  const float Pi=3.14;
   float R,S


  R=1.5;
  S=Pi*R*R;
  printf("S=%f\n",S);

}
```

Структура программы на языках Pascal и C++

Pascal

Program имя;

Uses <список модулей>

Раздел описаний

Begin

Раздел операторов

End.

C++

Директивы препроцессора

Описания глобальных объектов

Функции, написанные программистом

void main()

{

Операторы функции main

}

Pascal

- **Заголовок программы** начинается с зарезервированного слова Program и содержит имя программы, которое дает сам программист.
- **Подключение модулей** начинается с зарезервированного слова Uses и содержит список модулей (библиотек внешних процедур и функций).
- **Раздел описаний** состоит из подразделов, в которых объявляются все встречающиеся в программе объекты: метки (Label), константы (Const), типы (Type), переменные (Var), внутренние процедуры (Procedure), внутренние функции (Function). Описания отделяются друг от друга точкой с запятой.
- **Раздел операторов** заключается в операторные скобки **Begin** и **End**, при этом после **End** ставится **точка**. Операторы отделяются друг от друга точкой с запятой.

C++

- **Директивы препроцессора** включают в себя необходимые заголовочные файлы библиотек C++, определения констант, типов и макроопределений, используемых в программе.
- **Глобальные объекты** – константы, типы, переменные используются во всех функциях программы, обеспечивая соответствие типов, значений констант и переменных во всей программе.
- **Функции** – это особым образом оформленные части программы, которая выполняют действия необходимые программисту.
- **Функция main( ) -** главная функция программы. Любая программа на C++ обязательно включает в себя функцию main( ), с которой и начинает свое выполнение. Функция main() запускается операционной системой. Слово void перед именем функции main() означает, что функция main() не возвращает операционной системе никакой информации.
- **Тело функции** заключено в фигурные скобки ‘ **{ }** ’ . Фигурные скобки ‘ { } ’ обозначают начало и конец составного оператора (аналогично **begin** и **end** в TurboPascal). Точка с запятой ‘ ; ‘ завершает каждый оператор и каждое описание.
- Программа может содержать **комментарии**, их можно вставлять в любое место программы, где допускаются пробелы или в конце строки. В C++ используются два вида комментариев:
- /* многострочный
- комментарий */
- // однострочный комментарий до конца текущей строки

Среда программирования – это **интегрированная среда разработки программ (ИСРП)**, которая содержит:

- **редактор текста** - для создания и редактирования текста программы на языке высокого уровня, т.е. формирования исходного модуля (например, среды на основе языка Pascal сохраняют файл с расширением **.pas**; на основе языка C++ с расширением **.cpp**);

- **компилятор** - для перевода текста программы с языка высокого уровня в машинные коды, т.е. формирование объектного модуля (например, в Pascal **.tpu;** в **C++** **.obj**);

- **компоновщик** - для подключения объектных кодов стандартных команд и формирования загрузочного модуля (файл с расширением **.exe**)

- **загрузчик** - для выполнения загрузочного модуля программы.

Функции ИСРП

ИСРП позволяет:

1) создавать и редактировать исходные тексты программ;
2) сохранять исходные тексты программ в файлах;
3) считывать файлы с диска;
4) осуществлять поиск и исправление ошибок (отладка);
5) выполнять программу и просматривать результаты выполнения.

Рекомендации

1. Не следует стремиться к изучению как можно большего числа языков программирования.
2. Владеть дюжиной языков невозможно: их можно *знать,* но *знать* и *владеть* – не одно и тоже!
3. Изучив один язык, вы освоите главное – идеологию программирования. При необходимости – легко перейдете на другой язык.
4. Для продуктивной работы надо довести до автоматизма:
   - навыки правильного написания команд,
   - освоить множество функций, соглашений, умолчаний и др. тонкостей, характерных для данного языка.