**Информатика. Упражнение 6
Три уровня языков программирования**

**Цель работы:** познакомиться с различиями в программирования в машинных кодах, на языке ассемблера и на языке высокого уроня.

Чаще всего программы пишутся на языках высокого уровня, реже - на языках ассемблера и в исключительных случаях (обычно для управляющих ЭВМ) - в машинных кодах, но на ЭВМ выполняются только программы в машинных кодах. Перед выполнением программа должна быть странслирована (переведена) с языка высокого уровня или с языка ассемблера в машинные коды. Необходимым условием высокой квалификации программиста является понимание принципов выполнения программы на ЭВМ.

**Задание**

1. Изучите следующие разделы описания упражнения 6:
	* [структура ЭВМ](http://i.voenmeh.ru/kafi5/Kam.loc/inform/UprINF_6.html#struct);
	* [система команд ЭВМ](http://i.voenmeh.ru/kafi5/Kam.loc/inform/UprINF_6.html#kod);
	* [форматы чисел](http://i.voenmeh.ru/kafi5/Kam.loc/inform/UprINF_6.html#form);
	* [язык ассемблера](http://i.voenmeh.ru/kafi5/Kam.loc/inform/UprINF_6.html#ass);
	* [язык высокого уровня](http://i.voenmeh.ru/kafi5/Kam.loc/inform/UprINF_6.html#vis).
2. Возьмите у преподавателя номер [задачи](http://i.voenmeh.ru/kafi5/Kam.loc/inform/UprINF_6.html#zad).
3. Запрограммируйте задачу на всех трёх языках

**Cтруктура ЭВМ**

Рассмотрим простейшую гипотетическую ЭВМ, структура которой изображенна на рис. 1. ЭВМ состоит из процессора, оперативной памяти, устройсва ввода (клавиатуры) и устройсва вывода (дисплея).

|  |
| --- |
| http://i.voenmeh.ru/kafi5/Kam.loc/inform/EVM.png |
| Рис. 1. Структура ЭВМ |

Все числа на рисунке - в 16-й системе счисления.

**Система команд**

В отличие от большинства современных ЭВМ адреса в нашей ЭВМ имеют не байты, а 36-разрядные ячейки памяти. Команды ЭВМ имеют следующую структуру:

|  |  |  |  |
| --- | --- | --- | --- |
|

|  |  |  |
| --- | --- | --- |
|    |   |   |   |   |    |    |   |   |   |   |   |   |   |   |   |   |   |   |   |    |    |   |   |   |   |   |   |   |   |   |   |   |   |   |    |

 |
|

|  |  |  |
| --- | --- | --- |
|  0     КОп     5 |  6               Адрес 1                     25 |  26              Адрес 2                     35 |

 |

Адреса в команде - 15-разрядные. Максимальный адрес - 215-1 = 7FFF16 = 3276710. Максимальное количество команд - 26 = 6410. Для выполнения упражнения понадобятся следующие 16 команд.

***Арифметические операции***

Результат арифметической операции остаётся в АЛУ на сумматоре.

|  |  |  |
| --- | --- | --- |
| 01 |    A1    |    A2    |

- сложение с плавающей запятой операндов с адресами A1 и A2

|  |  |  |
| --- | --- | --- |
| 02 |    A1    |    A2    |

- вычитание с плавающей запятой из операнда с адресом A1 операнда с адресом A2

|  |  |  |
| --- | --- | --- |
| 03 |    A1    |    A2    |

- умножение с плавающей запятой операндов с адресами A1 и A2

|  |  |  |
| --- | --- | --- |
| 04 |    A1    |    A2    |

- деление с плавающей запятой операнда с адресами A1 на операнд с адресом A2

|  |  |  |
| --- | --- | --- |
| 05 |    A1    |    A2    |

- сложение с фиксированной запятой операндов с адресами A1 и A2

|  |  |  |
| --- | --- | --- |
| 06 |    A1    |    A2    |

- вычитание с фиксированной запятой из операнда с адресом A1 операнда с адресом A2

|  |  |  |
| --- | --- | --- |
| 07 |    A1    |    A2    |

- умножение с фиксированной запятой операндов с адресами A1 и A2

|  |  |  |
| --- | --- | --- |
| 08 |    A1    |    A2    |

- деление с фиксированной запятой операнда с адресами A1 на операнд с адресом A2;
      остаток от деления отбрасывается

***Операции пересылки***

|  |  |  |
| --- | --- | --- |
| 10 |    A1    |    A2    |

- пересылка из ячейки с адресом A1 в ячейку с адресом A2

|  |  |  |
| --- | --- | --- |
| 11 |    A1    |             |

- пересылка из сумматора в ячейку с адресом A1.

***Операции перехода***

|  |  |  |
| --- | --- | --- |
| 12 |    A1    |             |

- безусловный переход в ячейку с адресом A1.

|  |  |  |
| --- | --- | --- |
| 13 |    A1    |    A2    |

- условный переход. Если содержимое сумматора <0, то - по адресу A1, иначе переход по адресу A2.

***Операции ввода-вывода***

|  |  |  |
| --- | --- | --- |
| 77 |    A1    |             |

- ввод с клавиатуры в ячейку с адресом A1 целого числа.

|  |  |  |
| --- | --- | --- |
| 78 |    A1    |             |

- ввод с клавиатуры в ячейку с адресом A1 числа с плавающей запятой.

|  |  |  |
| --- | --- | --- |
| 87 |    A1    |             |

- вывод на экран дисплея из ячейки с адресом A1 целого числа.

|  |  |  |
| --- | --- | --- |
| 88 |    A1    |             |

- вывод на экран дисплея из ячейки с адресом A1 числа с плавающей запятой.

В реальной ЭВМ алгоритмы ввода и ввода реализуются с помощью сложных аппаратных средств и специального программного обеспечения (драйверов и подсистемы ввода-вывода, входящей в состав операционной системы). Для упрощения выполнения упражнения будем считать, что ввод или вывод содержимого ячейки делается одной машинной командой. Пользователь может вводить с клавиатуры только шестнадцатиричные цифры, т.е. символы 1, 2, . . , 10, А, B, . . ,F, знак *минус* и точку. На экран дисплея содержимое ячейки памяти выводится в шестнадцатиричной системе счисления.

На рис. 1 в оперативной памяти записаны программа и данные для вычисления выражения:

 e = (a + b) \* (c + d),

 где a=1502, b=498, c=53, d=47 в дес. системе.

Предполагается, что значения a, b, c и d предворительно записаны в память.

**Форматы чисел**

***Формат числа с фиксированной запятой (целого)***

|  |  |  |
| --- | --- | --- |
|

|  |  |
| --- | --- |
| ± |    |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |    |

 |
|

|  |  |
| --- | --- |
| 0 |   1                                                                                                                              35 |

 |

Максимальное целое число равно 235-1 = 34 359 738 36710 = 7FF FFF FFF16 .

***Формат числа с плавающей запятой***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|

|  |  |  |  |
| --- | --- | --- | --- |
| ± |    |   |   |   |   |   |    | ± |    |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |    |

 |
|

|  |  |  |  |
| --- | --- | --- | --- |
| 0 |   1                     7  | 8 |                                                                                                   35 |

 |

В разряды 0 - 7 записывается порядок числа. В разряде 0 - знак порядка.

В разряды с 8-го по 35-й помещается мантисса. В 8-м разряде - знак мантиссы.

**Язык ассемблера**

Программирование в машинных кодах очень трудоёмко. Программа в кодах трудно читается, в ней легко сделать ошибку. Кроме того, возникает задача распределения памяти, которая на языках более высокого уровня выполняется транслятором. Программа на языке ассемблера - это тоже последовательность машинных команд, только коды команд заменены сокращёнными названиями, а адреса - идентификаторами переменных. Например, команда сложения целочисленных переменных a и b на языке ассемблера записывается так:

 AddC a b

В табл.1 дан список команд ассемблера, соответствующих описанным выше машинным кодам.

|  |
| --- |
| **Табл. 1. Команды ассемблера** |
| **№п.п.** | **Имя команды** | **Операнды** | **Описание команды** | **Пример** |
| **Арифметические** |
| 1 | AddP | Операнд1 Операнд2 | Сложение чисел с пл. запятой. Результат - на сумматоре | AddP a b |
| 2 | DedP | Операнд1 Операнд2 | Вычитание чисел с пл. запятой. Результат - на сумматоре | DedP a b |
| 3 | MulP | Операнд1 Операнд2 | Умножение чисел с пл. запятой. Результат - на сумматоре | MulP a b |
| 4 | DivP | Операнд1 Операнд2 | Деление чисел с пл. запятой. Результат - на сумматоре | DivP a b |
| 5 | AddC | Операнд1 Операнд2 | Сложение целых чисел. Результат - на сумматоре | AddC a b |
| 6 | DedC | Операнд1 Операнд2 | Вычитание целых чисел. Результат - на сумматоре | DedC a b |
| 7 | MulC | Операнд1 Операнд2 | Умножение целых чисел. Результат - на сумматоре | MulC a b |
| 8 | DivC | Операнд1 Операнд2 | Деление целых чисел. Результат - на сумматоре | DivC a b |
| **Команды пересылки** |
| 9 | Send | Операнд1 Операнд2 | Пересылка: Операнд1 → Операнд2Эту команду можно использовать для присвоения переменной значения константы | Send a bSend 5 a |
| 10 | SendS | Операнд1 | Пересылка: Сумматор → Операнд1 | SendS a |
| **Команды перехода** |
| 11 | Go | Метка1 | Безусловный переход к команде с меткой | Go M1. . .M1: AddC a b |
| 12 | GoIf | Метка1 Метка2 | Условный переход. Если содержимое сумматора <0,то - Метка1, иначе Метка2. | GoIf M1 M2M1: AddC a b . . . M2: AddC d e |
| **Команды ввода-вывода** |
| 13 | RC | Операнд1 | Ввод с клавиатуры в Операнд1 целого числа. | RC a |
| 14 | RP | Операнд1 | Ввод с клавиатуры в Операнд1 числа с плавающей запятой. | RP a |
| 15 | WrC | Операнд1 | Вывод на экран дисплея Операнда1 целого типа. | WrC a |
| 16 | WrP | Операнд1 | Вывод на экран дисплея Операнда1 с плавающей запятой. | WrP a |

**Язык высокого уровня**

Для выполнения упражнения понадобится кроме машинных команд и языка ассемблера упрощённый язык высокого уровня. Опишем его.

В описании языка будем использовать два обозначения

 ::= - "это есть";

 | - "или".

 Пример

 знак числа ::= + | -

 Читается так: *знак числа это есть плюс или минус.*

**Алфавит:** латинские буквы, арабские цифры.

**Комментарий** начинается символами // и распространяется до конца строки.

**Константы** - десятичные числа.

**Идентификаторы переменных**. Все переменные делятся на скаляры и массивы. Идентификатор переменной начинается с буквы и состоит из букв и цифр.

Элемент массива обозначается так:

Идентификатор\_переменной[номер элемента]

Примеры:

 Ar1[5]

 E[0]

*Идентификатор\_переменной[номер элемента]* будем называть идентификатором элемента массива.

**Операции**
Арифметические: +, -, \*, /.
Сравнения: ==, >, <, <=, >=.

***Операторы языка***

**Операторы объявления переменных**

Целого типа

 int идентификатор\_скаляра;

 int идентификатор\_массива(длина массива);

Вещественного типа

 real идентификатор скаляра;

 real идентификатор массива(длина массива);

Примеры

 int a; //скаляр *a* целого типа

 real Ar(10); //массив Ar из десяти элементов

**Оператор присваивания**

 Идентификатор\_скаляра | Идентификатор элемента массива = арифметическое выражение;

Арифметическое выражение может состоять из одного идентификатора переменной.

Примеры

 a = Ar[2];

 f = (a + b) \* (c + d);

 F[4] = Ar[3];

**Оператор условия**

 if(условие)

 блок операторов

или

 if(условие)

 блок операторов

 else

 блок операторов

**Блок операторов** ::= оператор | последовательность операторов, заключённая в фигурные скобки.

**Условие** ::= арифметическое\_выражение значок\_операции\_cравнения арифметическое\_выражение

 Пример

 if(a > b)

 c = 1;

 else

 { c = 2;

 d = c +e;

 }

**Операторы цикла**

Оператор *while* - делать, пока выполняется условие

 while(условие)

 блок операторов

Пример

//Подсчитывается сумма арифметической прогрессии

 N = 1;

 s = 0;

 while(N <= 10)

 { s = s + N;

 N = N + 1;

 }

 // s = 55

Оператор *for* - делать, заданное число раз.

 for(начальное значение переменной цикла; условие выхода из цикла; изменение переменной цикла)

 блок операторов

 Пример

 //Подсчитывается сумма арифметической прогрессии

 s = 0;

 for( i=1; i <= 10; i = i +1 )

 { s = s + 1;

 }

 // s = 55

**Операторы ввода-вывода**

Оператор *read* - ввод с клавиатуры

 read Идентификатор\_скаляра | Идентификатор элемента массива;

 Примеры

 read a;

 read Ar[2];

Оператор *write* - вывод на экран

 write Идентификатор\_скаляра | Идентификатор элемента массива;

 Примеры

 write a;

 write Ar[2];

**Пример программы на трёх языках**

***Задача.*** Вводятся 10 натуральных чисел. Найти среди них наибольшее и наименьшее

***Решение в машиных кодах***

|  |  |  |
| --- | --- | --- |
| **Адрес ячейкипамяти** | **Данные или команда** | **Комментарии** |
| 000) | 9 | К - количество вводимых чисел минус 1 |
| 001) | 0 | Nmin |
| 002) | 0 | Nmax |
| 003) |   | Ni - текущее введённое число |
| 004) | 1 | шаг цикла |
| . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . |   |
| 008) | 77 | 001 | 000 | Ввод начального значения Nmin |
| 009) | 10 | 001 | 002 | Начальное значение Nmax = Nmin |
| 00A) | 77 | 003 | 000 | Ввод Ni |
| 00B) | 06 | 003 | 001 | Ni - Nmin |
| 00C) | 13 | 011 | 00D | Если Ni < Nmin, то перейти по адресу *011* |
| 00D) | 06 | 003 | 002 | Ni - Nmax |
| 00E) | 13 | 012 | 00F | Если Ni > Nmax, то перейти по адресу *00F* |
| 00F) | 10 | 003 | 002 | Nmax = Ni |
| 010) | 12 | 012 | 000 | Перейти в 012) |
| 011) | 10 | 003 | 001 | Nmin = Ni |
| 012) | 06 | 000 | 004 | АЛУ = K-1 |
| 013) | 10 | 000 | 000 | K - АЛУ - уменьшить счётчик |
| 014) | 13 | 015 | 00A | Если K >= 0, то перейти по адресу *00A* |
| 015) | 88 | 001 | 000 | Вывод на экран Nmin |
| 016) | 88 | 002 | 000 | Вывод на экран Nmax |
| . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . |   |

***Решение на языке ассемблера***

 Send 9 K //K = 9 количество вводимых чисел минус 1

 RC Nmin //Ввод начального значения Nmin

 Send Nmin Nmax //Начального значения Nmax = Nmin

 Send 1 i //i = 1 шаг цикла

 M5: RC N //Ввод N

 DedC N Nmin //N - Nmin

 GoIf M1 M2 //Если N < Nmin, то перейти к M1

 M2: DedC N Nmax //N - Nmax

 GoIf M3 M6 //Если N <= Nmax, то перейти к M3

 M6: Send N Nmax //Nmax = N

 Go M3

 M1: Send N Nmin //Nmin = N

 M3: DedC K i // Сумматор = K - i

 Send K //K = Сумматор

 GoIf M4 M5 //Если K >= 0, то перейти к M5

 M4: WrC Nmin

 WrC Nmax // Вывод Nmax

***Решение на языке высокого уровня***

 int Nmax;

 int Nmin;

 int N;

 int i;

 read Nmax;

 Nmin = Nmax;

 for (i = 1;i<=10;i = i + 1)

 { read N;

 if(N < Nmin) Nmin = N;

 else if (N > Nmax) Nmax = N;

 }

 write Nmin;

 write Nmax;

#### *Задача 10*

Ввести натуральное число *a*. Определить, в какой из следующих диапазонов оно попадает:

 1. 0 ≤ a ≤ 10 ;

 2. 10 < a ≤ 100 ;

 3. 100 < a ≤ 1000 ;

 4. a > 1000 .