**МИНОБРНАУКИ РОССИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕНЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.Шухова»**

**(БГТУ им. В.Г.Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа № 1

По дисциплине: Алгоритмы и СД

Тема: «Встроенные структуры данных(Pascal/C)»

Выполнил: ст.группы ПВ - 21

Браткова И.О.

Белгород 2017

**Цель работы:** изучение базовых типов данных языка Pascal/C как структур данных (СД).

**Вариант 4**

З а д а н и е

1. Для типов данных (см. Варианты заданий в таблицах 1,2) определить:

1.1. Абстрактный уровень представления СД:

1.1.1. Характер организованности и изменчивости.

1.1.2. Набор допустимых операций.

1.2. Физический уровень представления СД:

1.2.1. Схему хранения.

1.2.2. Объем памяти, занимаемый экземпляром СД.

1.2.3. Формат внутреннего представления СД и способ его интерпретации.

1.2.4. Характеристику допустимых значений.

1.2.5. Тип доступа к элементам.

1.3. Логический уровень представления СД.

Способ описания СД и экземпляра СД на языке программирования.

2. Для заданных типов данных определить набор значений, необходимый для изучения физического уровня представления СД.

3. Преобразовать значения в двоичный код.

4. Преобразовать двоичный код в значение.

5. Разработать и отладить программу, выдающую двоичное представление значений, заданных СД.

В программе использовать процедуры PrintByte и PrintVar.

Спецификация процедуры PrintByte:

1. Заголовок: procedure PrintByte(a:byte)/void PrintByte(unsigned char a).

2. Назначение: выводит на экран монитора двоичное представление переменной a типа byte/unsigned char.

3. Входные параметры: a.

4. Выходные параметры: нет.

Рекомендации: использовать побитовые операции сдвига и логического умножения.

Спецификация процедуры PrintVar:

1. Заголовок: procedure PrintVar(var a; size:word)/ void PrintVar(void a, unsigned int size).

2. Назначение: выводит на экран монитора двоичное представление переменной a произвольного типа размером size байт.

3. Входные параметры: a — переменная произвольного типа, значение которой выводится на экран в двоичном представлении (нетипизованный параметр); size — объем памяти (в байтах) занимаемый переменной a.

4. Выходные параметры: нет.

Рекомендации: нетипизованную переменную a привести к типу «массив байт», значение каждого элемента которого выводить на экран в двоичном представлении процедурой PrintByte.

6. Обработать программой значения, полученные в результате выполнения пункта 3 задания. Сделать выводы.

7. Разработать и отладить программу, определяющую значение переменной по ее двоичному представлению по следующему алгоритму:

1. Ввести двоичный код в переменную S строкового типа.

2. Преобразовать S в вектор B типа «массив байт».

3. Привести B к заданному типу. Вывести значение.

4. Конец.

8. Обработать программой значения, полученные в результате выполнения пункта 4 задания. Сделать выводы.

**СИ**

**Тип: unsigned char**

1. Абстрактный уровень представления СД:
   1. Характер организованности и изменчивости: **последовательный, статический.**
   2. Набор допустимых операций: **доступ, инициализация, сравнение, присваивание, арифметические операции.**
2. Физический уровень представления СД:
   1. Схема хранения: **последовательная.**
   2. Объём памяти, занимаемой экземпляром СД: **1 байт.**
   3. Формат внутреннего представления СД и способ его интерпретации: **char 1 байт (8 бит), например число 7 = 00000111**
   4. Характеристика допустимых значений: **[0..255]**
   5. Тип доступа к элементам: **прямой**
3. Логический уровень представления СД:

3.1 Способ описания СД и экземпляра СД на языке программирования: **unsigned char a;**

**Тип: float**

1. Абстрактный уровень представления СД:
   1. Характер организованности и изменчивости. **Последовательный, статический.**
   2. Набор допустимых операций: **доступ, инициализация, сравнение, присваивание, арифметические операции.**
2. Физический уровень представления СД:
   1. Схема хранения: **последовательная.**
   2. Объём памяти, занимаемой экземпляром СД: **4 байт.**
   3. Формат внутреннего представления СД и способ его интерпретации: **значение типа float хранится в памяти как 4 байтное число со знаком. Условно можно разбить на поля:**

**Поле знака – 1 бит, поле мантиссы – 8 бит, поле порядка – 23 бит.**

s=1бит

m=23 бит

e=8 бит

* 1. Характеристика допустимых значений: **[1.1754943E–38...1.175494E+38]**
  2. Тип доступа к элементам: **прямой.**

1. Логический уровень представления СД:
   1. Способ описания СД и экземпляра СД на языке программирования: **float a;**

**Тип: float массив [10][10]**

1. Абстрактный уровень представления СД:
   1. Характер организованности и изменчивости: **линейная структура, статический.**
   2. Набор допустимых операций: **доступ, присваивание.**
2. Физический уровень представления СД:
   1. Схема хранения: **последовательная.**
   2. Объём памяти, занимаемой экземпляром СД: **V = Vэл\*k = 40 байт, 4 байт занимает элемент массива.**
   3. Формат внутреннего представления СД и способ его интерпретации: **последовательность элементов типа float, хранящаяся, как одномерный массив. Номер элемента в массиве i\*n+j, где i – номер строки, j – столбца, а n – размерность (10\*10 в данном случае).**
   4. Характеристика допустимых значений: **для элементов - [1.1754943E–38...1.175494E+38]**
   5. Тип доступа к элементам: **прямой.**
3. Логический уровень представления СД:
   1. Способ описания СД и экземпляра СД на языке программирования: **float a[10][10];**

**Задание 2:**

1. Unsigned char: 15;
2. Float: -23,57;
3. Float [5][5]

|  |  |
| --- | --- |
| -23,57 | 1,99 |
| 4,22 | -2,88 |

**Задание 3:**

Тип 1. 00000000 00000000 00000000 00001111

Тип 2. 11000001 10111110 00000000 00000000

Тип 3. 11000001 10111110 00000000 00000000 00111111 11111110 10111000 01010010

01000000 10000111 00001010 00111101 11000000 00111000 01010001 11101100

**Задание 4:**

1. Unsigned char: 15;
2. Float: -23,57;
3. Float [5][5]

|  |  |
| --- | --- |
| -23,57 | 1,99 |
| -4,22 | 2,88 |

**Задание 5:**

#include <stdio.h>

#include <stdlib.h>

void PrintByte (unsigned char a)

{

unsigned int m = 128, i=0;

unsigned char t = a;

for (i=0; i<8; i++)

{

printf("%d", (t&m)?1:0);

t<<=1;

}

}

void PrintVar(void \*a, unsigned int size)

{

char i, j;

unsigned char \*bit = (unsigned char\*)a;

for (i=size-1; i>=0; i--) PrintByte(bit[i]);

bit = NULL;

}

void main()

{

float a=-23.75;

PrintVar(&a, sizeof(float));

char x=15;

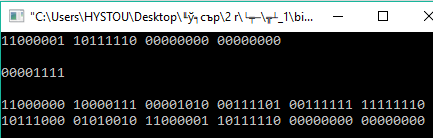
PrintVar(&x, sizeof(char));

float f[2][2] = {-23.75, 1.99, -4.22, 2.88};

PrintVar(&f, 4\*sizeof(float));

}

**Задание 6:**



**Задание 7:**

int conv\_i (char \*s)

{

char len = strlen(s);

char size = len/8;

char a[size];

int\* b;

int mask=1, i=0, j=0, st=0;

for(i=size-1; i>=0; i--)

{

for(j=0; j<8; j++)

{

a[i]<<=1;

if (s[st++]=='1')

a[i]|=mask;

}

}

b = (int\*)a;

return \*b;

}

float conv\_f(char \*s)

{

char len = strlen(s);

char size = len/8;

char a[size];

float\* b;

int mask=1, i=0, j=0, st=0;

for(i=size-1; i>=0; i--)

{

for(j=0; j<8; j++)

{

a[i]<<=1;

if (s[st++]=='1')

a[i]|=mask;

}

}

b = (float\*)a;

return \*b;

}

void main()

{

char \*s1 = "000000000000000000001111";

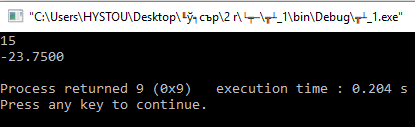
char \*s2 = "11000001101111100000000000000000";

printf ("%i\n", conv\_i(s1));

printf ("%.4f\n", conv\_f(s2));

}

**Задание 8**

****