МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Белгородский государственный технологический университет им. В.Г. Шухова»

**ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И УПРАВЛЯЮЩИХ СИСТЕМ Кафедра программного обеспечения вычислительной техники и автоматизированных систем**

Лабораторная работа №1

по дисциплине: Алгоритмы и Структуры Данных

Работу выполнил:

Студент группы ПВ-21:

Донцов Александр Алексеевич

Проверил:

Синюк Василий Григорьевич

Белгород – 2018

Л а б о р а т о р н а я р а б о т а № 1

**Встроенные структуры данных(Pascal/С)**

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

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.

Выполнение

Вариант 1

**Задание 1**

|  |  |  |
| --- | --- | --- |
| int | float | {red, yellow, green}colors |

**int**

1. Абстрактный уровень представления СД
   1. Характер организованности и изменчивости:

Целый упорядоченный тип, простейший

* 1. Набор допустимых операций:

- Операции доступа

- Битовые операции

- Логические операции

- Операции присваивания

- Операции сравнения

- Арифметические операции

2. Физический уровень представления СД:

2.1 Схема хранения:

Вырожденная СД (тип)

2.2 Объём памяти, занимаемый экземпляром СД:

(2/4/8) байт в зависимости от операционной системы (длинна машинного слова)

2.3 Формат внутреннего представления СД и способ его интерпретации:

простой

2.4 Характеристика допустимых значений

−32768, +32767

2.5 Тип доступа к элементам

Прямой

3.Логический уровень представления СД

Способ описания СД – int

Способ описания экземпляра СД на языке программирования

- int a;

**float**

1.Абстрактный уровень представления СД

1.1 Характер организованности и изменчивости:

Вещественный тип, неупорядочен, простейший

1.2 Набор допустимых операций:

- Операции доступа

- Операции инициализации

- Операции присваивания

- операции сравнения

- Арифметические операции

2. Физический уровень представления СД:

2.1 Схема хранения:

Последовательная (Вырожденная СД)

2.2 Объём памяти, занимаемый экземпляром СД:

4 байт

2.3 Формат внутреннего представления СД и способ его интерпретации:

1 бит – знак (1- “+”, 0 – “ -”)

5 бит – порядок

10 бит – мантиса

Формула для вычисления значения, хранящегося в памяти

v = (–1)s⋅2e-1023⋅1.m, если 0 < e < 2047

v = (–1)s⋅21022⋅0.m, если e = 0 и m ≠ 0

v = (–1)s, если e = 0 и m = 0

v = (–1)s⋅Inf, если e = 2047 и m = 0

v = NaN, если e = 2047 и m ≠ 0

Точность 15 — 16 знаков.

2.4 Характеристика допустимых значений

-2 147 483 648.0  / 2 147 483 647.0

2.5 Тип доступа к элементам

Прямой

3.Логический уровень представления СД

Способ описания СД – float

Способ описания экземпляра СД на языке программирования

- float b;

**{red, yellow, green}colors**

1. Абстрактный уровень представления СД

1.1 Характер организованности и изменчивости:

Целый упорядоченный, статический

1.2 Набор допустимых операций:

- операция доступа

- операция инициализации(только при описании)

- операция сравнения

- Арифметические операции

2. Физический уровень представления СД:

2.1 Схема хранения: последовательная(вырожденная СД)

2.2 Объём памяти, занимаемый экземпляром СД:

2/4/8 байт

2.3 Формат внутреннего представления СД и способ его интерпретации:

первый в списке 0, остальные +1

2.4 Характеристика допустимых значений

-32768…+32767

2.5 Тип доступа к элементам

прямой

3.Логический уровень представления СД

Способ описания СД

Описание структуры:

Способ описания: **{red, yellow, green}colors**

Экземпляр СД: colors n = red

**Задание 2-4**

int a = 95;

float d = -1.34567;

typedef enum {

red = 1,

yellow,

green

} colors;

number = yellow;

int = 00100010

d = 1100000000111000000111101011100001010001111010111000010100011111

number = 000000000000000000000000000010

**Задание 5**

// выводит на экран монитора двоичное представление переменной а

//типа int

void printByte(unsigned char a) {

int i;

int mask = 1 << 7;

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

printf("%i" , (a & mask) ? 1 : 0);

a <<= 1;

}

}

//выводит на экран монитора двоичное представление переменной a

//произвольного типа размера size байт

void printVar(void \* a, size\_t size) {

int i;

char \* p = (char \*) a;

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

printByte(\*(p + i));

printf("\n");

}



**Задание 7**

//вводит с консоли символы в строку s и возвращает количество прочитанных символов

int inputBits(int s[]) {

int count = 0;

char c;

while ((c = getchar()) != '\n')

\*(s + count++) = c;

\*(s + count) = '\0';

return count;

}

//преобразовывает последовательностей нулей и единиц в строке s размером size в последовательность бит

// возвращает не типизированный указатель

void \*transfVar(char \*s, int size) {

int sizeOfByts = size / 8;

char \* rez =(char \*) malloc(sizeOfByts);

int i, j, pos;

for (i = 0; i < sizeOfByts; i++) {

pos = sizeOfByts - i - 1;

rez[pos] = 0;

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

rez[pos] = rez[pos] << 1;

rez[pos] = rez[pos] | (s[i \* 8 + j] == '1' ? 1 : 0);

}

}

return rez;

}



**Код всей программы**

#include "stdafx.h"

#include "malloc.h"

// выводит на экран монитора двоичное представление переменной а

//типа int

void printByte(char);

//выводит на экран монитора двоичное представление переменной a

//произвольного типа размера size байт

void printVar(void \*, size\_t);

//вводит с консоли символы в строку s и возвращает количество прочитанных символов

int inputBits(char []);

//преобразовывает последовательностей нулей и единиц в строковой константе в последовательность бит

//возврщает нетипизированный указатель

void \*transfVar(char \*, int);

int main() {

int c = 90;

float = -1.34567;

typedef enum {

red = 1,

yellow,

green

} colors;

number = five;

printf("i = %i ", c);

printVar(&c, sizeof(c));

printf("f = %.7f ", d);

printVar(&d, sizeof(d));

printf("enum ");

printVar(&number, sizeof(number));

char s[100];

int size = inputBits(s);

int char \*b = (int \*) transfVar(s, size);

printf("%d ",\*b);

int t;

scanf("%d", &t);

return 0;

}

int inputBits(char s[]) {

int count = 0;

char c;

while ((c = getchar()) != '\n')

\*(s + count++) = c;

\*(s + count) = '\0';

return count;

}

void \*transfVar(char \*s, int size) {

int sizeOfByts = size / 8;

char \* rez =(char \*) malloc(sizeOfByts \* sizeof(char));

int i, j, pos;

for (i = 0; i < sizeOfByts; i++) {

pos = sizeOfByts - i - 1;

rez[pos] = 0;

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

rez[pos] = rez[pos] << 1;

rez[pos] = rez[pos] | (s[i \* 8 + j] == '1' ? 1 : 0);

}

}

return rez;

}

void printByte(char a) {

int i;

char mask = 1 << 7;

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

printf("%i" , (a & mask) ? 1 : 0);

a <<= 1;

}

}

void printVar(void \* a, size\_t size) {

int i;

char \* p = (char \*) a;

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

printByte(\*(p + i));

printf("\n");}