МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ 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 структуры данных тема: «Встроенные структуры данных»

Выполнил: студент группы ПВ-223 Мелехов Артём Дмитриевич

Проверил:

асс. Солонченко Роман Евгеньевич

Лабораторная работа №1 «Встроенные структуры данных»

Цель работы: Изучение базовых типов данных языка С как структур данных (СД).

Содержание отчета:

- Тема лабораторной работы;
- Цель лабораторной работы;
- Условия задач и их решение;
- Вывод.

Задания к лабораторной работе: вариант №7

```
long
double(comp)
typedef enum Animal {
    cat,
    dog,
    mouse,
    tiger
} Animal; (animal(cat, dog, mouse, tiger))
```

1. Для типов данных определить

(а) Абстрактный уровень представления СД

Для типа данных long:

- Абстрактный уровень представления: целое число, представленное в двоичном формате со знаком.
- Характер организованности: последовательный набор битов, где каждый бит представляет собой степень двойки.
- Изменчивость: значение числа может изменяться путем присваивания нового значения переменной.
- Набор допустимых операций: арифметические операции (+, -, *, /, %), операции сравнения (==, !=, <, >, <=, >=), операции побитового сдвига (<<, >>), побитовые операции $(\&, |, ^)$, присваивание (=), инкремент (++), декремент (--), условный оператор (?:).

Для типа данных double:

- Абстрактный уровень представления: вещественное число, представленное в формате с плавающей точкой.
- Характер организованности: число представлено в виде мантиссы и порядка, где мантисса представляет собой дробное число в двоичном формате, а порядок определяет местоположение запятой.

- Изменчивость: значение числа может изменяться путем присваивания нового значения переменной.
- Набор допустимых операций: арифметические операции (+, -, *, /), операции сравнения (==, !=, <, >, <=, >=), присваивание (=), инкремент (++), декремент (--), условный оператор (?:).

Для перечисления Animal:

- Абстрактный уровень представления: набор констант, представляющих животных.
- Характер организованности: каждая константа имеет свой уникальный целочисленный код.
- Изменчивость: перечисление не может изменяться во время выполнения программы.
- Набор допустимых операций: присваивание (=), сравнение (==, !=), доступ к элементам перечисления.

(b) Физический уровень представления СД:

Для типа данных long:

- Схема хранения: последовательный набор битов, где каждый бит представляет собой степень двойки.
 - Объем памяти: 8 байт (64 бита).
- Формат внутреннего представления: двоичное число со знаком, где старший бит отводится под знак.
 - Допустимые значения: от -9,223,372,036,854,775,808 до 9,223,372,036,854,775,807.
 - Тип доступа к элементам: прямой доступ к значению переменной.

Для типа данных double:

- Схема хранения: формат с плавающей точкой IEEE 754.
- Объем памяти: 8 байт (64 бита).
- Формат внутреннего представления: мантисса и порядок, где мантисса представляет собой дробное число в двоичном формате, а порядок определяет местоположение запятой.
 - Допустимые значения: от ~4.9 x 10^-324 до ~1.8 x 10^308.
 - Тип доступа к элементам: прямой доступ к значению переменной.

Для перечисления Animal:

- Схема хранения: каждая константа имеет свой уникальный целочисленный код.

- Объем памяти: зависит от количества констант в перечислении.
- Формат внутреннего представления: целочисленные коды для каждой константы.
- Допустимые значения: константы, определенные в перечислении.
- Тип доступа к элементам: прямой доступ к константам перечисления.
- (с) Логический уровень представления СД

```
long name; -long;

double name; -comp;

typedef enum Animal {
    cat,
    dog,
    mouse,
    tiger
} Animal; -animal(cat, dog, mouse, tiger)
```

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

```
typedef enum Animal {
    cat,
    dog,
    mouse,
    tiger
} Animal;

int main() {
    long a_max = 57;
    long a_min = -41;
    double b = 12.5;
    enum Animal c = cat, d = dog, e = mouse, f = tiger;
    return 0;
}
```

3. Преобразовать значения в двоичный код

$$a_{max}=57$$
1. Делаем деление на 2: 57 / 2 $=28$ (остаток 1)

- 2. Записываем остаток в конец двоичного кода: 1
- 3. Делаем деление на 2: 28 / 2 = 14 (остаток 0)

- 4. Записываем остаток в конец двоичного кода: 01
 - 5. Делаем деление на 2: 14/2 = 7 (остаток 0)
- 6. Записываем остаток в конец двоичного кода: 001
 - 7. Делаем деление на 2: 7/2 = 3 (остаток 1)
- 8. Записываем остаток в конец двоичного кода: 1001
 - 9. Делаем деление на 2: 3/2 = 1 (остаток 1)
- 10. Записываем остаток в конец двоичного кода: 11001
 - 11. Делаем деление на 2: 1/2 = 0 (остаток 1)
- 12. Записываем остаток в конец двоичного кода: 111001
- 13. Дописываем в начало знаковый бит и незначащие нули: 0'0000000000000000000000111001

Otbet: $57_{10} = 0'000000000000000000000000111001_2$

$$a_{min} = -41$$

- 1. Находим абсолютное значение числа: |(-41)| = 41
 - 2. Делаем деление на 2: 41 / 2 = 20 (остаток 1)
 - 3. Записываем остаток в конец двоичного кода: 1
 - 4. Делаем деление на 2: 20 / 2 = 10 (остаток 0)
 - 5. Записываем остаток в конец двоичного кода: 01
 - 6. Делаем деление на 2: 10 / 2 = 5 (остаток 0)
- 7. Записываем остаток в конец двоичного кода: 001
 - 8. Делаем деление на 2: 5 / 2 = 2 (остаток 1)
- 9. Записываем остаток в конец двоичного кода: 1001
 - 10. Делаем деление на 2: 2 / 2 = 1 (остаток 0)
- 11. Записываем остаток в конец двоичного кода: 01001
 - 12. Делаем деление на 2: 1 / 2 = 0 (остаток 1)
- 13. Записываем остаток в конец двоичного кода, добавляем знаковый бит и незначащие нули:

1'00000000000000000000000000101001

- - 15. Полученный результат является двоичным кодом числа 41.

- 1. Определяем знак числа: положительное.
 - 2. Определяем целую часть числа: 12.
 - 3. Определяем дробную часть числа: 0.5.
- 4. Преобразуем целую часть в двоичный код: 12/2 = 6 (остаток 0), 6/2
 - = 3 (остаток 0), 3 / 2 = 1 (остаток 1), 1 / 2
 - = 0 (остаток 1). Целая часть числа 12 в двоичном коде равна 1100.
- 5. Преобразуем дробную часть в двоичный код с помощью умножения на 2 и записи целой части до тех пор, пока не получим ноль или не достигнем максимального количества знаков после запятой. $0.5\,*\,2$
- = 1 (целая часть), оставшаяся дробная часть равна 0.0, процесс завершен. Дробная часть числа 12.5 в двоичном коде равна 0.1.
 - 6. Объединяем двоичные коды целой и дробной частей, разделяя их точкой: 1100.1.
 - 7. Полученный результат является двоичным кодом числа 12.5.

- 4. Преобразовать двоичный код в значение
 - 1. Это положительное число (первый бит равен 0).
 - 2. Значащие биты: 000000000000000000000111001. Инвертируем их:

11111111111111111111111111000110.

- 4. Переводим в десятичную систему счисления: 57.
- 1. Определить знак числа: отрицательное (первый бит равен 1).
- 2. Инвертировать значащие биты, оставив первый бит неизменным:

- 3. Добавить 1 к полученному числу: 1'1111111111111111111111111011000.
 - 4. Перевести полученное число в десятичную систему счисления: -40.
- 5. Полученное число равно значению исходного отрицательного двоичного кода:

1. Знак числа равен 0, значит число положительное.

- 2. Порядок равен 0110, что в двоичной системе счисления равно 6. Так как формат числа предполагает смещенный порядок, то необходимо вычесть из него 0111 (бинарное представление числа 7) и получить 1 в десятичной системе счисления. Значит порядок равен 2^{-1} = 0.5.

системе счисления равно 1.5 в десятичной системе счисления.

- 4. Итоговое значение равно: $(-1)^0 * 2^{0.5} * 1.5 = 12.5$.
- 5. Разработать и отладить программу, выдающую двоичное представление значений, заданных СД

```
#include <stdio.h>
typedef enum Animal {
    cat, dog, mouse, tiger
} Animal;
void print byte(unsigned char a) {
    for (int i = 7; i >= 0; i--) {
        printf("%d", (a >> i) & 1);
    printf(" ");
}
void print var(void *a, unsigned size) {
    unsigned char *p = a;
    for (int i = size - 1; i >= 0; i--) {
        print byte(*(p + i));
    printf("\n");
}
int main() {
    long 1 = 57;
    long m = -41;
    double d = 3.14159265358979323846;
    Animal animal = tiger;
    printf("Binary representation of long l:\n");
    print var(&l, sizeof(l));
    printf("Binary representation of long m:\n");
    print var(&m, sizeof(m));
```

```
printf("Binary representation of double d:\n");
print_var(&d, sizeof(d));

printf("Binary representation of Animal animal:\n");
print_var(&animal, sizeof(animal));

return 0;
}
```

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

Выводы: в ходе выполнения работы были изучены базовые типы данных языка С как СД.