**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационных систем**

отчет

**по практической работе №1**

**по дисциплине «Программирование»**

Тема: Типы данных и их внутреннее представление в памяти

|  |  |  |
| --- | --- | --- |
| Студент(ка) гр. |  | Гечис В.Р. |
| Преподаватель |  | Глущенко А.Г. |

Санкт-Петербург

2022

**Цель работы.**

Изучение и организация типов данных; получение практических навыков работы с типами данных; определение внутреннего представления типов данных в памяти.

**Основные теоретические положения.**

Память компьютера можно представить в виде непрерывной последовательности двоичных ячеек, каждая из которых может находиться в двух состояниях условно обозначаемых 0 и 1. Каждая такая двоичная ячейка называется **битом**. Вся эта последовательность ячеек условно разбита на порции из 8 бит, называемые **байтами**. Таким образом, 1 байт = 8 битам. Байт является основной единицей измерения объема памяти.

С каждым байтом памяти связано понятие **адреса**, который, по сути, является номером байта в непрерывной последовательности байтов памяти компьютера. То есть каждый байт памяти имеет свой адрес. По этому адресу и осуществляется доступ к данным, хранящимся в памяти.

Тип данных для каждого программного объекта, представляющего данные, определяет:

·         характер данных (число, со знаком или без знака, целое или с дробной частью, одиночный символ или текст, представляющий последовательность символов и т.д.);

·         объем памяти, который занимают в памяти эти данные;

·         диапазон или множество возможных значений;

·         правила обработки этих данных (например, допустимые операции).

Внутреннее представление величин целого типа – целое число в двоичном коде. При использовании спецификатора signed старший бит числа интерпретируется как знаковый (0 – положительное число, 1 – отрицательное). Для кодирования целых чисел со знаком применяется прямой, обратный и дополнительный коды.

Представление положительных и отрицательных чисел в прямом, обратном и дополнительном кодах отличается. В прямом коде в знаковый разряд помещается цифра 1, а в разряды цифровой части числа – двоичный код его абсолютной величины.

Увидеть, каким образом тип данных представляется на компьютере, можно при помощи логических операций: побитового сдвига (<<) и поразрядной конъюнкции (&).

**Постановка задачи.**

Необходимо написать единую программу, объединяющую последовательное выполнение всех заданий практической работы.

Используя побитовое И, и сдвиг вывести двоичное представление целочисленного типа данных int.

Организовать переход между блоками программы.

Используя объединение типа float и int вывести по аналогичному алгоритму двоичное представление типа double.

Используя объединение типа double и массива из двух целочисленных значений int.

**Выполнение работы.**

Код программы представлен в приложении А.

/\*

Блок описания кода и использованных алгоритмов

\*/

/\*

Блок скриншотов работы программы

\*/

/\*

Блок таблицы с тестовыми данными

\*/

**Выводы.**

В ходе данной лабораторной работы были изучены типы данных и их внутреннее представление в памяти компьютера.

Приложение А

рабочий код

#include <iostream>  
#include <string>  
  
using namespace std;  
int main() {  
 // 1 razdel pracc PAB0Tbl  
  
 cout << "int:" << sizeof(int) << "\n" << "short int:" << sizeof(short int) << "\n" << "long int:"  
 << sizeof(long int) << "\n" << "float:" << sizeof(float) << "\n" << "double:" << sizeof(double) << "\n"  
 << "long double:" << sizeof(long double) << "\n" << "char:" << sizeof(char) << "\n" << "bool:" << sizeof(bool)  
 << "\n";  
 //2 razdel  
  
 int binaryInt;  
 cout << "Enter data of type int:\n";  
 while (!(cin >> binaryInt)) {  
 cout << "error, expecting a num value\n";  
 cin.clear();  
 fflush(stdin);  
 }  
 int b = sizeof(binaryInt) \* 8 - 1;  
 int maskInt = 1 << b;  
 int arrInt[32];  
 for (int i = 0; i <= b; i++) {  
 if (i == 1 || i % 8 == 0) {  
 putchar(' ');  
 }  
  
 putchar(binaryInt & maskInt ? '1' : '0');  
 if (binaryInt & maskInt){  
 arrInt[i]=1;  
 }  
 else {  
 arrInt[i]=0;  
  
 }  
  
 binaryInt = binaryInt << 1;  
 }  
  
 cout << "\nDo you want to shift all the bits equal to one to the left?\n'1'- Yes '2' - No\n";  
 int userChoiceInt;  
 while (!(cin >> userChoiceInt)) {  
 cout << "error, expecting a num value\n";  
 cin.clear();  
 fflush(stdin);  
 }  
 if (userChoiceInt != 1 && userChoiceInt !=2){  
 cout << "error, expecting '1' or '2' value:\n";  
 cin >> userChoiceInt;  
 }  
 if (userChoiceInt == 1) {  
  
  
 if (arrInt[0]==1){  
 arrInt[0]=0;  
 }  
  
 for (int g=1; g<32;g++) {  
 if (arrInt[g] == 1) {  
 arrInt[g] = 0;  
 arrInt[g-1] = 1;  
 }  
 }  
// for ( int l=0;l<32;l++){  
// cout << arrInt[l];  
// }  
 for ( int c =0;c<32;c++){  
 if (c == 1 || c % 8 == 0) {  
 putchar(' ');  
 }  
 cout<<arrInt[c];  
 }  
  
  
 }  
  
  
  
 int d = 0;  
 do {  
  
 int Type;  
 cout << "\n Select type:\nfloat - 1 \ndouble - 2 \n";  
 while (!(cin >> Type)) {  
 cout << "error, expecting '1' or '2' value\n";  
 cin.clear();  
 fflush(stdin);  
 }  
 if (Type != 1 && Type !=2){  
 cout << "error, expecting '1' or '2' value:\n";  
 cin >> Type;  
 }  
 if (Type == 1) {  
 //3 razdel  
 cout << "\n";  
 cout << "Enter data of type float: \n";  
 union {  
 float binaryFloat;  
 int value;  
 };  
 unsigned order = sizeof(float) \* 8, mask = 1 << order - 1;  
 int arrFloat[32];  
 while (!(cin >> binaryFloat)) {  
 cout << "error, expecting num value\n";  
 cin.clear();  
 fflush(stdin);  
 }  
 for (int i = 0; i <= order-1; ++i) {  
 if (i == 1 || i == 9) {  
 putchar(' ');  
 }  
 putchar(value & mask ? '1' : '0');  
  
  
 if (value & mask){  
 arrFloat[i]=1;  
 }  
 else {  
 arrFloat[i]=0;  
 }  
 value <<= 1;  
  
 }  
 cout << endl;  
 cout << "\nDo you want to shift all the bits equal to one to the left?\n'1'- Yes '2' - No\n";  
 int userChoiceFloat;  
 while (!(cin >> userChoiceFloat)) {  
 cout << "error, expecting a num value\n";  
 cin.clear();  
 fflush(stdin);  
 }  
 if (userChoiceFloat != 1 && userChoiceFloat !=2){  
 cout << "error, expecting '1' or '2' value:\n";  
 cin >> userChoiceInt;  
 }  
 if (userChoiceFloat == 1) {  
  
 if (arrFloat[0]==1){  
 arrFloat[0]=0;  
 }  
  
 for (int g=1; g<32;g++) {  
 if (arrFloat[g] == 1) {  
 arrFloat[g] = 0;  
 arrFloat[g-1] = 1;  
 }  
 }  
  
 for ( int c =0;c<32;c++){  
 if (c == 1 || c == 9) {  
 putchar(' ');  
 }  
 cout<<arrFloat[c];  
 }  
  
 }  
 }  
  
  
 if (Type == 2) {  
 //4 razdel  
 cout << "\n";  
 cout << "Enter data of type double: \n";  
 union {  
 double binaryDouble;  
 int arr[2];  
 };  
 while (!(cin >> binaryDouble)) {  
 cout << "error, expecting num value\n";  
 cin.clear();  
 fflush(stdin);  
 }  
 int arrDouble[64];  
 int orderg = sizeof(double) \* 8;  
 int gmaskk = 1 << (orderg - 1);  
 for (int j = 1; j >= 0; j--) {  
 for (int i = 0; i <= (orderg / 2)-1; i++)  
 {  
 putchar(arr[j] & gmaskk ? '1' : '0');  
 if (j==1) {  
  
 if (arr[j] & gmaskk) {  
 arrDouble[i] = 1;  
 } else {  
 arrDouble[i] = 0;  
 }  
 }  
 if (j==0){  
 if (arr[j] & gmaskk) {  
 arrDouble[i+32] = 1;  
 } else {  
 arrDouble[i+32] = 0;  
  
 }  
  
 }  
 arr[j] <<= 1;  
  
 if ((i == 0 || i == 11) && j == 1) {  
 putchar(' ');  
  
 }  
 }  
 }  
 cout << "\nDo you want to shift all the bits equal to one to the left?\n'1'- Yes '2' - No\n";  
 int userChoiceDouble;  
 while (!(cin >> userChoiceDouble)) {  
 cout << "error, expecting a num value\n";  
 cin.clear();  
 fflush(stdin);  
 }  
 if (userChoiceDouble != 1 && userChoiceDouble !=2){  
 cout << "error, expecting '1' or '2' value:\n";  
 cin >> userChoiceInt;  
 }  
 if (userChoiceDouble == 1) {  
  
 if (arrDouble[0]==1){  
 arrDouble[0]=0;  
 }  
  
 for (int g=1; g<64;g++) {  
 if (arrDouble[g] == 1) {  
 arrDouble[g] = 0;  
 arrDouble[g-1] = 1;  
 }  
 }  
  
 for ( int c =0;c<64;c++){  
 if (c == 1 || c == 12) {  
 putchar(' ');  
 }  
 cout<<arrDouble[c];  
 }  
  
 }  
  
 }  
  
 cout << "\nDo you want to exit, or you want to continue? \n1 - continue 2 - exit\n";  
 cin >> d;  
 } while (d!=2);  
  
}

Выполеннеие работы

|  |  |
| --- | --- |
| Меню | |
| При запуске программы перед пользователем выводится кол-во байтов, занимаемое каждым типом данных и предлагается ввести значение типа int для вывода его двоичного представления | Меню:    Проверка на ввод символов, которые не входят в диапазон выбора: |

Продолжение Таблицы

|  |  |
| --- | --- |
| После ввода значение, предлагается выполнить модификацию из ИДЗ, с проверкой на введённое значение, определяющее переход, затем предлагается выбрать, какой вещественный тип данных представить далее. | После определения типа данных, выводится его двоичное представление, предлагается выполнить модификацию. |

Продолжение Таблицы

|  |  |
| --- | --- |
| После выполнения всех блоков, пользователю предлагается либо повторить выбор, начиная с выбора типа данных, либо закрыть программу. |  |