федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет «Московский институт электронной техники»

Направление подготовки 09.03.04 «Программная инженерия»

Дисциплина «Электротехника»

**Лабораторная РАБОТА №1**

**Представление данных в ЭВМ**

Работу выполнил студенты группы ПИН-24 Баранов Д.А. и Демочкина А.В.

Работу проверил ассистент Института СПИНТех Фомин Р.А.

**Цель работы:** изучить размеры стандартных типов C/C++ и форматы представления чисел на выбранной платформе.

**Вариант:**

****

**Задание №1.**

Выберите платформу, на которой планируется в дальнейшем выполнять лабораторные работы.

Для выполнения заданий выбран онлайн компилятор <https://godbolt.org/>.

ОС и разрядность ОС: GNU/Linux 64

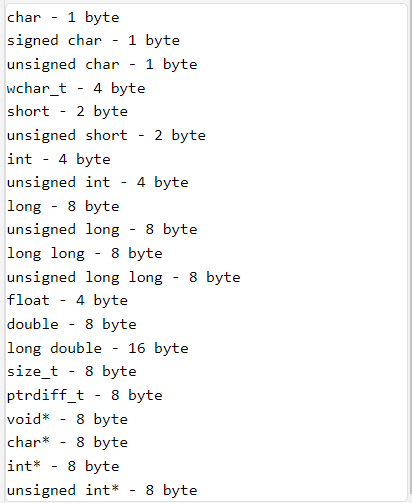
Компилятор: Compiler Explorer x86-64 gcc 11.2

Архитектура: x86-64

**Задание №2.**

При помощи оператора sizeof языка С/С++ выясните, сколько байтов занимают переменные следующих типов языка С/С++: char, signed char, unsigned char, wchar\_t, short, unsigned short, int, unsigned int, long, unsigned long, long long, unsigned long long, float, double, long double, size\_t, ptrdiff\_t, void\*, char\*, int\*, unsigned int\* на выбранной платформе.





#include <iostream>

using namespace std;

int main()

{

char i = '1';

cout<<"char - " << sizeof(i)<<" byte "<<endl;

cout<<"signed char - "<<sizeof((signed char)i)<<" byte "<<endl;

cout<<"unsigned char - "<<sizeof((unsigned char)i)<<" byte "<<endl;

cout<<"wchar\_t - "<<sizeof((wchar\_t)i)<<" byte "<<endl;

cout<<"short - "<<sizeof((short)i)<<" byte "<<endl;

cout<<"unsigned short - "<<sizeof((unsigned short)i)<<" byte "<<endl;

cout<<"int - "<<sizeof((int)i)<<" byte "<<endl;

cout<<"unsigned int - "<<sizeof((unsigned int)i)<<" byte "<<endl;

cout<<"long - "<<sizeof((long)i)<<" byte "<<endl;

cout<<"unsigned long - "<<sizeof((unsigned long)i)<<" byte "<<endl;

cout<<"long long - "<<sizeof((long long)i)<<" byte "<<endl;

cout<<"unsigned long long - "<<sizeof((unsigned long long)i)<<" byte "<<endl;

cout<<"float - "<<sizeof((float)i)<<" byte "<<endl;

cout<<"double - "<<sizeof((double)i)<<" byte "<<endl;

cout<<"long double - "<<sizeof((long double)i)<<" byte "<<endl;

cout<<"size\_t - "<<sizeof((size\_t)i)<<" byte "<<endl;

cout<<"ptrdiff\_t - "<<sizeof((ptrdiff\_t)i)<<" byte "<<endl;

void\* p;

cout<<"void\* - "<<sizeof(p)<<" byte "<<endl;

cout<<"char\* - "<<sizeof((char\*)p)<<" byte "<<endl;

cout<<"int\* - "<<sizeof((int\*)p)<<" byte "<<endl;

cout<<"unsigned int\* - "<<sizeof((unsigned int\*)p)<<" byte "<<endl;

}

Целочисленные типы short, unsigned short имеют разрядность 16 бит.

Целочисленные типы int, unsigned int имеют разрядность 32 бита.

Целочисленные типы long, unsigned long, long long, unsigned long long имеют разрядность 64 бита.

Типы с плавающей запятой разрядность 16 бит не имеют.

Тип с плавающей запятой float имеет разрядность 32 бита.

Тип с плавающей запятой double имеет разрядность 64 бита.

**Задание №3.**

Изучите, как интерпретируется одна и та же область памяти, если она рассматривается как знаковое или беззнаковое целое число, а также — как одно и то же число записывается в различных системах счисления.

Для этого на языке C/C++ разработайте функцию void 𝑝𝑟𝑖𝑛𝑡16(void \* 𝑝), которая печатает для 16-битной области памяти по заданному адресу 𝑝:

а) целочисленную беззнаковую интерпретацию в шестнадцатеричном представлении;

б) целочисленную беззнаковую интерпретацию в двоичном представлении;

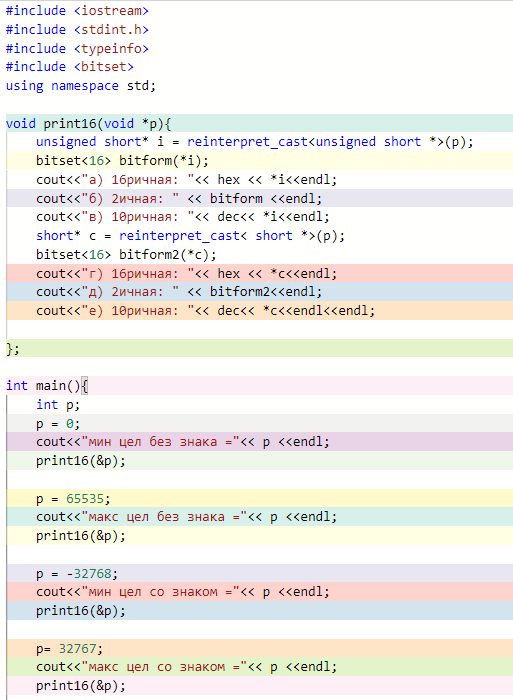
в) целочисленную беззнаковую интерпретацию в десятичном представлении;

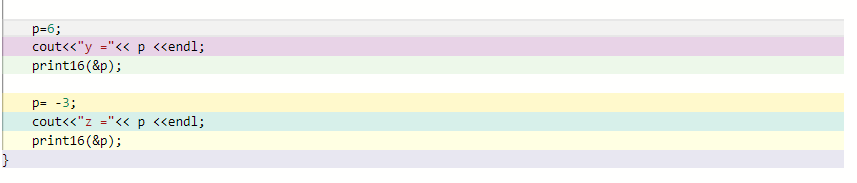
г) целочисленную знаковую интерпретацию в шестнадцатеричном представлении;

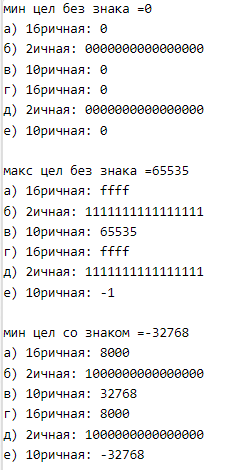
д) целочисленную знаковую интерпретацию в двоичном представлении;

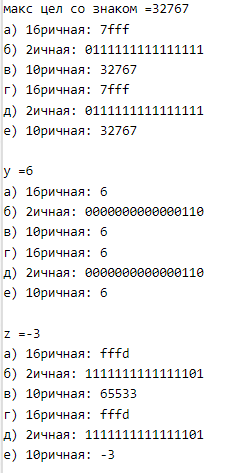
е) целочисленную знаковую интерпретацию в десятичном представлении.

|  | мин цел без знака =0 | макс цел без знака =65535 | мин цел со знаком =-32768 | макс цел со знаком =32767 | у =6 | z =-3 |
| --- | --- | --- | --- | --- | --- | --- |
| а) | 0 | ffff | 8000 | 7fff | 6 | fffd |
| б) | 0000000000000000 | 1111111111111111 | 1000000000000000 | 0111111111111111 | 0000000000000110 | 1111111111111101 |
| в) | 0 | 65535 | 32768 | 32767 | 6 | 65533 |
| е) | 0 | -1 | -32768 | 32767 | 6 | -3 |









#include <iostream>

#include <stdint.h>

#include <typeinfo>

#include <bitset>

using namespace std;

void print16(void \*p){

unsigned short\* i = reinterpret\_cast<unsigned short \*>(p);

bitset<16> bitform(\*i);

cout<<"a) 16ричная: "<< hex << \*i<<endl;

cout<<"б) 2ичная: " << bitform <<endl;

cout<<"в) 10ричная: "<< dec<< \*i<<endl;

short\* c = reinterpret\_cast< short \*>(p);

bitset<16> bitform2(\*c);

cout<<"г) 16ричная: "<< hex << \*c<<endl;

cout<<"д) 2ичная: " << bitform2<<endl;

cout<<"е) 10ричная: "<< dec<< \*c<<endl<<endl;

};

int main(){

int p;

p = 0;

cout<<"мин цел без знака ="<< p <<endl;

print16(&p);

p = 65535;

cout<<"макс цел без знака ="<< p <<endl;

print16(&p);

p = -32768;

cout<<"мин цел со знаком ="<< p <<endl;

print16(&p);

p= 32767;

cout<<"макс цел со знаком ="<< p <<endl;

print16(&p);

p=6;

cout<<"у ="<< p <<endl;

print16(&p);

p= -3;

cout<<"z ="<< p <<endl;

print16(&p);

}

В результате выполнения программы мы убедились, что шестнадцатеричное представление одно и тоже. Также двоичное представление одно и тоже.

**Задание №4.**

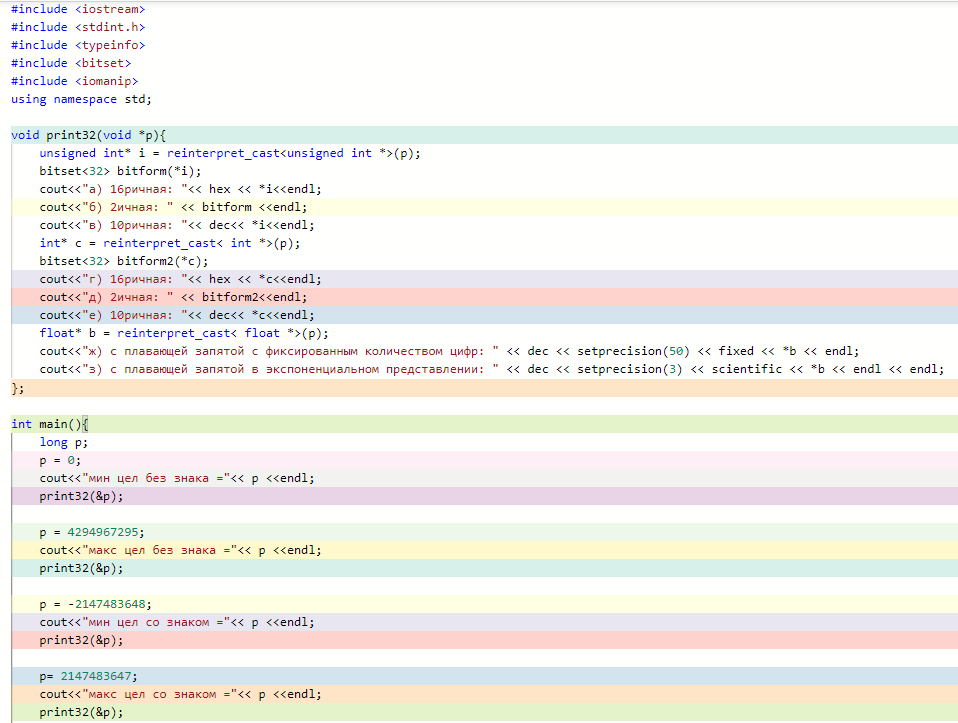
Разработайте на языке C/C++ функции 𝑝𝑟𝑖𝑛𝑡32() и 𝑝𝑟𝑖𝑛𝑡64(), аналогичные 𝑝𝑟𝑖𝑛𝑡16() для размеров 32 и 64 бита, и дополните их интерпретацией памяти как числа с плавающей запятой соответствующего размера. Необходимо напечатать:

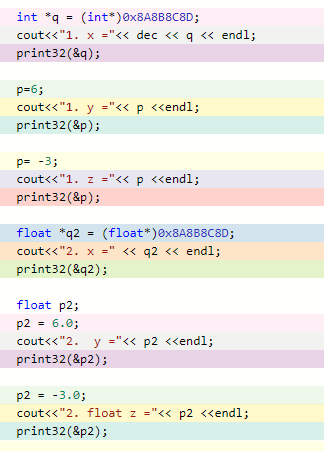
ж) интерпретацию с плавающей запятой в представлении с фиксированным количеством цифр после запятой;

з) интерпретацию с плавающей запятой в экспоненциальном представлении.

**Функция print32():**

|  | мин цел без знака =0 | макс цел без знака =4294967295 | мин цел со знаком =-2147483648 | макс цел со знаком =2147483647 | x = 0x8a8b8c8d | у =6 | z =-3 | float x = 0x8a8b8c8d | float у =6 | float z =-3 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| а) | 0 | ffffffff | 80000000 | 7fffffff | 8a8b8c8d | 6 | fffffffd | 8a8b8c8d | 40c00000 | c0400000 |
| б) | 00000000000000000000000000000000 | 11111111111111111111111111111111 | 10000000000000000000000000000000 | 01111111111111111111111111111111 | 10001010100010111000110010001101 | 00000000000000000000000000000110 | 11111111111111111111111111111101 | 10001010100010111000110010001101 | 01000000110000000000000000000000 | 11000000010000000000000000000000 |
| в) | 0 | 4294967295 | 2147483648 | 2147483647 | 2324401293 | 6 | 4294967293 | 2324401293 | 1086324736 | 3225419776 |
| е) | 0 | -1 | -2147483648 | 2147483647 | -1970566003 | 6 | -3 | -1970566003 | 1086324736 | -1069547520 |
| ж) | 0.00000000000000000000000000000000000000000000000000 | -nan | -0.00000000000000000000000000000000000000000000000000 | nan | -0.00000000000000000000000000000001343808244128746009 | 0.00000000000000000000000000000000000000000000840779 | -nan | -0.00000000000000000000000000000001343808244128746009 | 6.00000000000000000000000000000000000000000000000000 | -3.00000000000000000000000000000000000000000000000000 |
| з) | 0.000e+00 | -nan | -0.000e+00 | nan | -1.344e-32 | 8.408e-45 | -nan | -1.344e-32 | 6.000e+00 | -3.000e+00 |





#include <iostream>

#include <stdint.h>

#include <typeinfo>

#include <bitset>

#include <iomanip>

using namespace std;

void print32(void \*p){

unsigned int\* i = reinterpret\_cast<unsigned int \*>(p);

bitset<32> bitform(\*i);

cout<<"a) 16ричная: "<< hex << \*i<<endl;

cout<<"б) 2ичная: " << bitform <<endl;

cout<<"в) 10ричная: "<< dec<< \*i<<endl;

int\* c = reinterpret\_cast< int \*>(p);

bitset<32> bitform2(\*c);

cout<<"г) 16ричная: "<< hex << \*c<<endl;

cout<<"д) 2ичная: " << bitform2<<endl;

cout<<"е) 10ричная: "<< dec<< \*c<<endl;

float\* b = reinterpret\_cast< float \*>(p);

cout<<"ж) с плавающей запятой с фиксированным количеством цифр: " << dec << setprecision(50) << fixed << \*b << endl;

cout<<"з) с плавающей запятой в экспоненциальном представлении: " << dec << setprecision(3) << scientific << \*b << endl << endl;

};

int main(){

long p;

p = 0;

cout<<"мин цел без знака ="<< p <<endl;

print32(&p);

p = 4294967295;

cout<<"макс цел без знака ="<< p <<endl;

print32(&p);

p = -2147483648;

cout<<"мин цел со знаком ="<< p <<endl;

print32(&p);

p= 2147483647;

cout<<"макс цел со знаком ="<< p <<endl;

print32(&p);

int \*q = (int\*)0x8A8B8C8D;

cout<<"1. x ="<< dec << q << endl;

print32(&q);

p=6;

cout<<"1. у ="<< p <<endl;

print32(&p);

p= -3;

cout<<"1. z ="<< p <<endl;

print32(&p);

float \*q2 = (float\*)0x8A8B8C8D;

cout<<"2. x =" << q2 << endl;

print32(&q2);

float p2;

p2 = 6.0;

cout<<"2. y ="<< p2 <<endl;

print32(&p2);

p2 = -3.0;

cout<<"2. float z ="<< p2 <<endl;

print32(&p2);

}

Мин цел без знака =0

a) 16ричная: 0

б) 2ичная: 00000000000000000000000000000000

в) 10ричная: 0

г) 16ричная: 0

д) 2ичная: 00000000000000000000000000000000

е) 10ричная: 0

ж) с плавающей запятой с фиксированным количеством цифр: 0.00000000000000000000000000000000000000000000000000

з) с плавающей запятой в экспоненциальном представлении: 0.000e+00

макс цел без знака =4294967295

a) 16ричная: ffffffff

б) 2ичная: 11111111111111111111111111111111

в) 10ричная: 4294967295

г) 16ричная: ffffffff

д) 2ичная: 11111111111111111111111111111111

е) 10ричная: -1

ж) с плавающей запятой с фиксированным количеством цифр: -nan

з) с плавающей запятой в экспоненциальном представлении: -nan

мин цел со знаком =-2147483648

a) 16ричная: 80000000

б) 2ичная: 10000000000000000000000000000000

в) 10ричная: 2147483648

г) 16ричная: 80000000

д) 2ичная: 10000000000000000000000000000000

е) 10ричная: -2147483648

ж) с плавающей запятой с фиксированным количеством цифр: -0.00000000000000000000000000000000000000000000000000

з) с плавающей запятой в экспоненциальном представлении: -0.000e+00

макс цел со знаком =2147483647

a) 16ричная: 7fffffff

б) 2ичная: 01111111111111111111111111111111

в) 10ричная: 2147483647

г) 16ричная: 7fffffff

д) 2ичная: 01111111111111111111111111111111

е) 10ричная: 2147483647

ж) с плавающей запятой с фиксированным количеством цифр: nan

з) с плавающей запятой в экспоненциальном представлении: nan

1. x =0x8a8b8c8d

a) 16ричная: 8a8b8c8d

б) 2ичная: 10001010100010111000110010001101

в) 10ричная: 2324401293

г) 16ричная: 8a8b8c8d

д) 2ичная: 10001010100010111000110010001101

е) 10ричная: -1970566003

ж) с плавающей запятой с фиксированным количеством цифр: -0.00000000000000000000000000000001343808244128746009

з) с плавающей запятой в экспоненциальном представлении: -1.344e-32

1. у =6

a) 16ричная: 6

б) 2ичная: 00000000000000000000000000000110

в) 10ричная: 6

г) 16ричная: 6

д) 2ичная: 00000000000000000000000000000110

е) 10ричная: 6

ж) с плавающей запятой с фиксированным количеством цифр: 0.00000000000000000000000000000000000000000000840779

з) с плавающей запятой в экспоненциальном представлении: 8.408e-45

1. z =-3

a) 16ричная: fffffffd

б) 2ичная: 11111111111111111111111111111101

в) 10ричная: 4294967293

г) 16ричная: fffffffd

д) 2ичная: 11111111111111111111111111111101

е) 10ричная: -3

ж) с плавающей запятой с фиксированным количеством цифр: -nan

з) с плавающей запятой в экспоненциальном представлении: -nan

2. x =0x8a8b8c8d

a) 16ричная: 8a8b8c8d

б) 2ичная: 10001010100010111000110010001101

в) 10ричная: 2324401293

г) 16ричная: 8a8b8c8d

д) 2ичная: 10001010100010111000110010001101

е) 10ричная: -1970566003

ж) с плавающей запятой с фиксированным количеством цифр: -0.00000000000000000000000000000001343808244128746009

з) с плавающей запятой в экспоненциальном представлении: -1.344e-32

2. y =6.000e+00

a) 16ричная: 40c00000

б) 2ичная: 01000000110000000000000000000000

в) 10ричная: 1086324736

г) 16ричная: 40c00000

д) 2ичная: 01000000110000000000000000000000

е) 10ричная: 1086324736

ж) с плавающей запятой с фиксированным количеством цифр: 6.00000000000000000000000000000000000000000000000000

з) с плавающей запятой в экспоненциальном представлении: 6.000e+00

2. float z =-3.000e+00

a) 16ричная: c0400000

б) 2ичная: 11000000010000000000000000000000

в) 10ричная: 3225419776

г) 16ричная: c0400000

д) 2ичная: 11000000010000000000000000000000

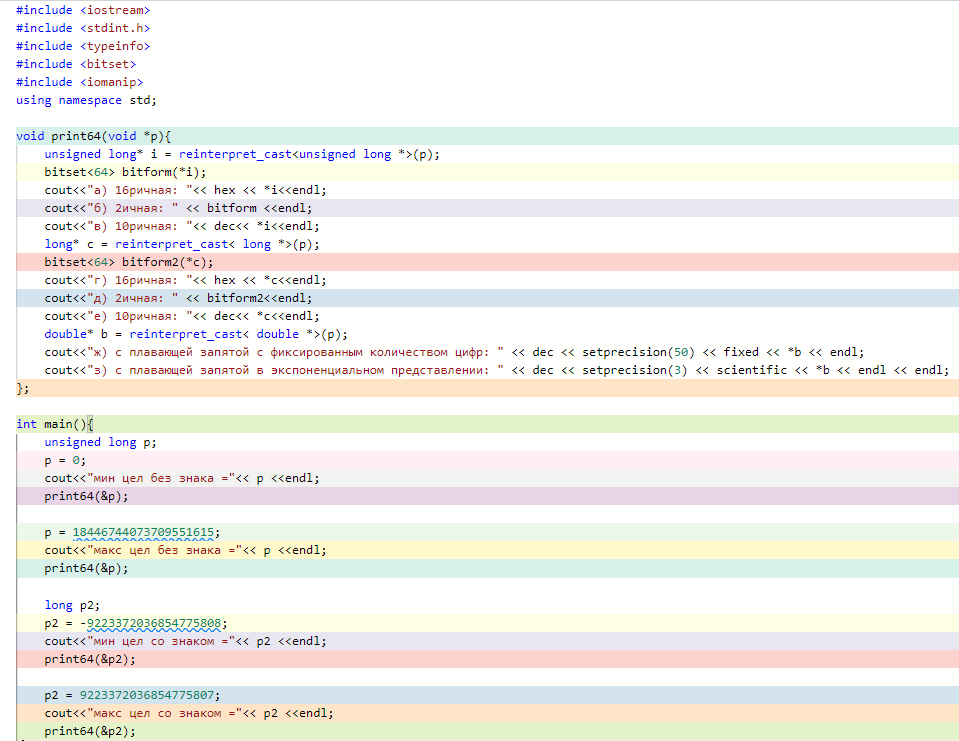
е) 10ричная: -1069547520

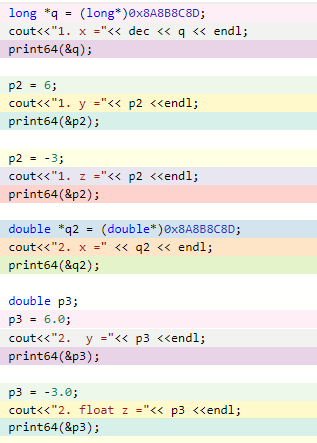
ж) с плавающей запятой с фиксированным количеством цифр: -3.00000000000000000000000000000000000000000000000000

з) с плавающей запятой в экспоненциальном представлении: -3.000e+00

**Функция print64():**

|  | мин цел без знака =0 | макс цел без знака =18446744073709551615 | мин цел со знаком =-9223372036854775808 | макс цел со знаком =9223372036854775807 | x = 0x8a8b8c8d | у =6 | z =-3 | double x = 0x8a8b8c8d | double у =6 | double z =-3 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| а) | 0 | ffffffffffffffff | 8000000000000000 | 7fffffffffffffff | 8a8b8c8d | 6 | fffffffffffffffd | 8a8b8c8d | 8a8b8c8d40c00000 | c008000000000000 |
| б) | 0000000000000000000000000000000000000000000000000000000000000000 | 1111111111111111111111111111111111111111111111111111111111111111 | 1000000000000000000000000000000000000000000000000000000000000000 | 0111111111111111111111111111111111111111111111111111111111111111 | 0000000000000000000000000000000010001010100010111000110010001101 | 0000000000000000000000000000000000000000000000000000000000000110 | 1111111111111111111111111111111111111111111111111111111111111101 | 0000000000000000000000000000000010001010100010111000110010001101 | 0100000000011000000000000000000000000000000000000000000000000000 | 1100000000001000000000000000000000000000000000000000000000000000 |
| в) | 0 | 18446744073709551615 | 9223372036854775808 | 9223372036854775807 | 2324401293 | 6 | 18446744073709551613 | 2324401293 | 4618441417868443648 | 13837309855095848960 |
| е) | 0 | -1 | -9223372036854775808 | 9223372036854775807 | 2324401293 | 6 | -3 | 2324401293 | 4618441417868443648 | -4609434218613702656 |
| ж) | 0.00000000000000000000000000000000000000000000000000 | -nan | -0.00000000000000000000000000000000000000000000000000 | nan | 0.00000000000000000000000000000000000000000000000000 | 0.00000000000000000000000000000000000000000000000000 | -nan | 0.00000000000000000000000000000000000000000000000000 | 6.00000000000000000000000000000000000000000000000000 | -3.00000000000000000000000000000000000000000000000000 |
| з) | 0.000e+00 | -nan | -0.000e+00 | nan | 1.148e-314 | 2.964e-323 | -nan | 1.148e-314 | 6.000e+00 | -3.000e+00 |





#include <iostream>

#include <stdint.h>

#include <typeinfo>

#include <bitset>

#include <iomanip>

using namespace std;

void print64(void \*p){

unsigned long\* i = reinterpret\_cast<unsigned long \*>(p);

bitset<64> bitform(\*i);

cout<<"a) 16ричная: "<< hex << \*i<<endl;

cout<<"б) 2ичная: " << bitform <<endl;

cout<<"в) 10ричная: "<< dec<< \*i<<endl;

long\* c = reinterpret\_cast< long \*>(p);

bitset<64> bitform2(\*c);

cout<<"г) 16ричная: "<< hex << \*c<<endl;

cout<<"д) 2ичная: " << bitform2<<endl;

cout<<"е) 10ричная: "<< dec<< \*c<<endl;

double\* b = reinterpret\_cast< double \*>(p);

cout<<"ж) с плавающей запятой с фиксированным количеством цифр: " << dec << setprecision(50) << fixed << \*b << endl;

cout<<"з) с плавающей запятой в экспоненциальном представлении: " << dec << setprecision(3) << scientific << \*b << endl << endl;

};

int main(){

unsigned long p;

p = 0;

cout<<"мин цел без знака ="<< p <<endl;

print64(&p);

p = 18446744073709551615;

cout<<"макс цел без знака ="<< p <<endl;

print64(&p);

long p2;

p2 = -9223372036854775808;

cout<<"мин цел со знаком ="<< p2 <<endl;

print64(&p2);

p2 = 9223372036854775807;

cout<<"макс цел со знаком ="<< p2 <<endl;

print64(&p2);

long \*q = (long\*)0x8A8B8C8D;

cout<<"1. x ="<< dec << q << endl;

print64(&q);

p2 = 6;

cout<<"1. у ="<< p2 <<endl;

print64(&p2);

p2 = -3;

cout<<"1. z ="<< p2 <<endl;

print64(&p2);

double \*q2 = (double\*)0x8A8B8C8D;

cout<<"2. x =" << q2 << endl;

print64(&q2);

double p3;

p3 = 6.0;

cout<<"2. y ="<< p3 <<endl;

print64(&p3);

p3 = -3.0;

cout<<"2. float z ="<< p3 <<endl;

print64(&p3);

}

Vин цел без знака =0

a) 16ричная: 0

б) 2ичная: 0000000000000000000000000000000000000000000000000000000000000000

в) 10ричная: 0

г) 16ричная: 0

д) 2ичная: 0000000000000000000000000000000000000000000000000000000000000000

е) 10ричная: 0

ж) с плавающей запятой с фиксированным количеством цифр: 0.00000000000000000000000000000000000000000000000000

з) с плавающей запятой в экспоненциальном представлении: 0.000e+00

макс цел без знака =18446744073709551615

a) 16ричная: ffffffffffffffff

б) 2ичная: 1111111111111111111111111111111111111111111111111111111111111111

в) 10ричная: 18446744073709551615

г) 16ричная: ffffffffffffffff

д) 2ичная: 1111111111111111111111111111111111111111111111111111111111111111

е) 10ричная: -1

ж) с плавающей запятой с фиксированным количеством цифр: -nan

з) с плавающей запятой в экспоненциальном представлении: -nan

мин цел со знаком =-9223372036854775808

a) 16ричная: 8000000000000000

б) 2ичная: 1000000000000000000000000000000000000000000000000000000000000000

в) 10ричная: 9223372036854775808

г) 16ричная: 8000000000000000

д) 2ичная: 1000000000000000000000000000000000000000000000000000000000000000

е) 10ричная: -9223372036854775808

ж) с плавающей запятой с фиксированным количеством цифр: -0.00000000000000000000000000000000000000000000000000

з) с плавающей запятой в экспоненциальном представлении: -0.000e+00

макс цел со знаком =9223372036854775807

a) 16ричная: 7fffffffffffffff

б) 2ичная: 0111111111111111111111111111111111111111111111111111111111111111

в) 10ричная: 9223372036854775807

г) 16ричная: 7fffffffffffffff

д) 2ичная: 0111111111111111111111111111111111111111111111111111111111111111

е) 10ричная: 9223372036854775807

ж) с плавающей запятой с фиксированным количеством цифр: nan

з) с плавающей запятой в экспоненциальном представлении: nan

1. x =0x8a8b8c8d

a) 16ричная: 8a8b8c8d

б) 2ичная: 0000000000000000000000000000000010001010100010111000110010001101

в) 10ричная: 2324401293

г) 16ричная: 8a8b8c8d

д) 2ичная: 0000000000000000000000000000000010001010100010111000110010001101

е) 10ричная: 2324401293

ж) с плавающей запятой с фиксированным количеством цифр: 0.00000000000000000000000000000000000000000000000000

з) с плавающей запятой в экспоненциальном представлении: 1.148e-314

1. у =6

a) 16ричная: 6

б) 2ичная: 0000000000000000000000000000000000000000000000000000000000000110

в) 10ричная: 6

г) 16ричная: 6

д) 2ичная: 0000000000000000000000000000000000000000000000000000000000000110

е) 10ричная: 6

ж) с плавающей запятой с фиксированным количеством цифр: 0.00000000000000000000000000000000000000000000000000

з) с плавающей запятой в экспоненциальном представлении: 2.964e-323

1. z =-3

a) 16ричная: fffffffffffffffd

б) 2ичная: 1111111111111111111111111111111111111111111111111111111111111101

в) 10ричная: 18446744073709551613

г) 16ричная: fffffffffffffffd

д) 2ичная: 1111111111111111111111111111111111111111111111111111111111111101

е) 10ричная: -3

ж) с плавающей запятой с фиксированным количеством цифр: -nan

з) с плавающей запятой в экспоненциальном представлении: -nan

2. x =0x8a8b8c8d

a) 16ричная: 8a8b8c8d

б) 2ичная: 0000000000000000000000000000000010001010100010111000110010001101

в) 10ричная: 2324401293

г) 16ричная: 8a8b8c8d

д) 2ичная: 0000000000000000000000000000000010001010100010111000110010001101

е) 10ричная: 2324401293

ж) с плавающей запятой с фиксированным количеством цифр: 0.00000000000000000000000000000000000000000000000000

з) с плавающей запятой в экспоненциальном представлении: 1.148e-314

2. y =6.000e+00

a) 16ричная: 4018000000000000

б) 2ичная: 0100000000011000000000000000000000000000000000000000000000000000

в) 10ричная: 4618441417868443648

г) 16ричная: 4018000000000000

д) 2ичная: 0100000000011000000000000000000000000000000000000000000000000000

е) 10ричная: 4618441417868443648

ж) с плавающей запятой с фиксированным количеством цифр: 6.00000000000000000000000000000000000000000000000000

з) с плавающей запятой в экспоненциальном представлении: 6.000e+00

2. float z =-3.000e+00

a) 16ричная: c008000000000000

б) 2ичная: 1100000000001000000000000000000000000000000000000000000000000000

в) 10ричная: 13837309855095848960

г) 16ричная: c008000000000000

д) 2ичная: 1100000000001000000000000000000000000000000000000000000000000000

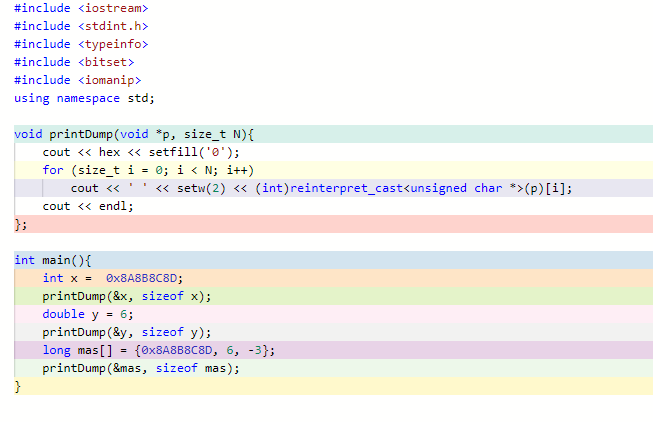
е) 10ричная: -4609434218613702656

ж) с плавающей запятой с фиксированным количеством цифр: -3.00000000000000000000000000000000000000000000000000

з) с плавающей запятой в экспоненциальном представлении: -3.000e+00

**Задание №5.**

Изучите, как располагаются в памяти байты, составляющие целое число и число с плавающей запятой. Для этого на языке C/C++ разработайте функцию void 𝑝𝑟𝑖𝑛𝑡𝐷𝑢𝑚𝑝(void \* 𝑝, size\_t 𝑁), которая печатает для области памяти по заданному адресу 𝑝 значения 𝑁 байтов, начиная с младшего, в шестнадцатеричном представлении (шестнадцатеричный дамп памяти).



#include <iostream>

#include <stdint.h>

#include <typeinfo>

#include <bitset>

#include <iomanip>

using namespace std;

void printDump(void \*p, size\_t N){

cout << hex << setfill('0');

for (size\_t i = 0; i < N; i++)

cout << ' ' << setw(2) << (int)reinterpret\_cast<unsigned char \*>(p)[i];

cout << endl;

};

int main(){

int x = 0x8A8B8C8D;

printDump(&x, sizeof x);

double y = 6;

printDump(&y, sizeof y);

long mas[] = {0x8A8B8C8D, 6, -3};

printDump(&mas, sizeof mas);

}

8d 8c 8b 8a

00 00 00 00 00 00 18 40

8d 8c 8b 8a 00 00 00 00 06 00 00 00 00 00 00 00 fd ff ff ff ff ff ff ff

Из результатов выполнения функции видно, что порядок следования байтов в словах для нашего процессора прямой (младший байт по младшему адресу).

**Вопросы**

1. Как представляются целые числа со знаком и без знака?

Беззнаковые целые числа представляются в виде двоичного представления. Либо в виде восьмеричной и шестнадцатеричной для более компактной записи двоичного кода.

Для чисел со знаком под шестнадцатеричным представлением числа подразумевается представление двоичного содержимого памяти, а не самих данных. Н-р, -1 = FFFF FFFF обозначает, что значение −1 представляется в памяти двоичным кодом 1111 1111 1111 1111 1111 1111 1111 1111, что в шестнадцатеричном виде соответствует FFFF FFFF.

Знаковые целые числа представляются в виде: величины со знаком, кода с избытком (какой-то избыток), дополнительного кода.

Величина со знаком: выделить один бит для хранения знака, а в оставшихся хранить абсолютную величину (модуль) числа. Соответственно, код в виде величины со знаком не используется для целых чисел. Идея раздельного кодирования знака и абсолютной величины используется при кодировании вещественных чисел с плавающей запятой.

Код с избытком: можно задать некоторую константу w и поставить в соответствие каждому знаковому числу 𝑥 беззнаковое значение 𝑢 = 𝑥 + w. Код с избытком используется для представления порядка вещественных чисел с плавающей запятой, а также в специальной аппаратуре или для передачи данных по каналам связи, если диапазон данных невелик, но заведомо несимметричен относительно нуля (в частности, год или температура в помещении в градусах Цельсия)

2. Как перевести число в дополнительный код?

Необходимо записывать отрицательное число −𝑥 тем кодом, который получается в результате беззнакового вычитания 0 − 𝑥 (с учётом цикличности сложения и вычитания в ЭВМ беззнаковое представление этого кода 2^𝑁 − 𝑥). Дополнительный код неотрицательных чисел совпадает с прямым (натуральным).

Дополнительный код −𝑥 может быть рассчитан как ¬𝑥+ 1, где ¬𝑥 — побитовое отрицание (инверсия битов) натурального двоичного представления абсолютной величины числа 𝑥.

Код позволяет заменить операцию вычитания на операцию сложения и сделать операции сложения и вычитания одинаковыми для знаковых и [беззнаковых](https://ru.wikipedia.org/wiki/%D0%91%D0%B5%D0%B7%D0%B7%D0%BD%D0%B0%D0%BA%D0%BE%D0%B2%D0%BE%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%BE) чисел.

Именно в дополнительном коде представлены отрицательные числа в современных вычислительных системах.

3. Для чего нужно знать порядок следования байтов на вашем компьютере?

Знание порядка байтов необходимо при передаче сведений, которые должны интерпретироваться сетью. Необходимо преобразовывать порядок данных, если компьютеры на обоих концах имеют разный порядок следования байтов, чтобы данные на обоих устройствах считывались одинаково.

Обратный порядок байтов позволяет легко читать числа, записанные в памяти. Обратный порядок принят в протоколе TCP/IP. Прямой порядок байтов удобен при обработке чисел большой разрядности с помощью процессора малой разрядности, так как позволяет при сложении таких чисел обращаться к памяти последовательно в порядке возрастания адресов, а такие запросы выполняются быстрее.

4. Как располагаются в памяти элементы массива?

Последовательно друг за другом, сначала представляется первый элемент массива, затем представляется второй и так далее.

5. Как найти размер массива, зная размер элемента и их количество?

Перемножить размер элемента с их количеством.