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 №2**

**Представление данных в ЭВМ. Сравнение платформ**

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

Работу проверил ассистент Института СПИНТех Фомин Р.А.

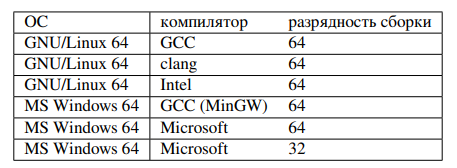
**Цель работы:** изучить размеры стандартных типов C/C++ и форматы представления чисел и символьной информации на различных платформах.

**Вариант 2**



**Задание 1**

Выполните измерения согласно заданию Л1.з2 на платформах, доступных на ВЦ (таблица Л2.1).





#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;

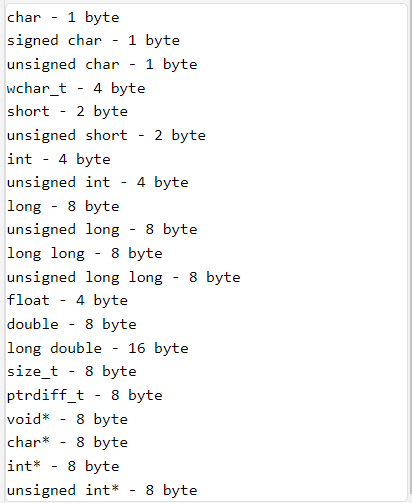
cout<<"int\* - "<<sizeof((int\*)p)<<" byte "<<endl;

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

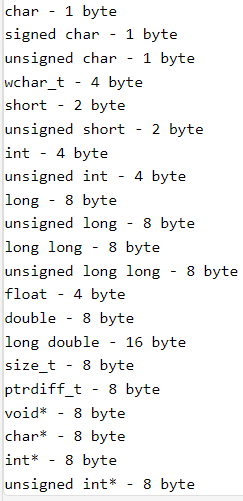
}

**GNU/Linux 64**

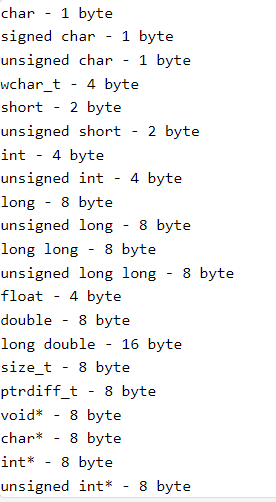
**GCC (x86-64 gcc 11.2):**



**clang (x86-64 clang 13.0.1):**

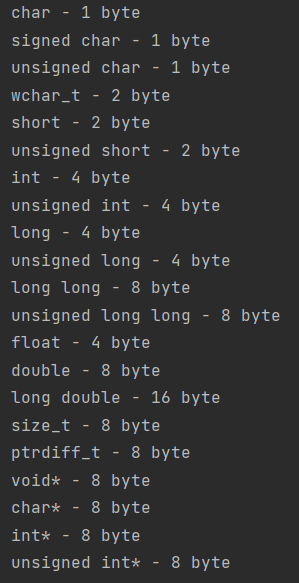


**ICC (x86-64 icc 2021.5.0):**

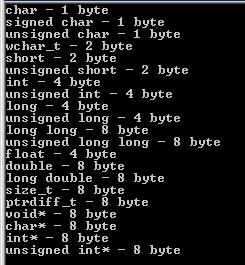
****

**MS Windows 64**

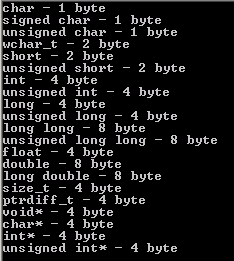
**GCC (MinGW):**



**Microsoft (x64):**

****

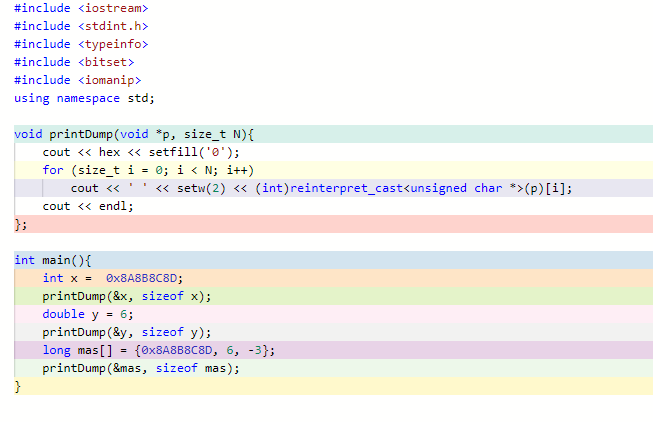
**Microsoft (x86):**



|  | **GCC (x86-64 gcc 11.2)** | **clang (x86-64 clang 13.0.1)** | **ICC (x86-64 icc 2021.5.0)** | **GCC (MinGW)** | **Microsoft (x64)** | **Microsoft (x86)** |
| --- | --- | --- | --- | --- | --- | --- |
| char | 1 | 1 | 1 | 1 | 1 | 1 |
| signed char | 1 | 1 | 1 | 1 | 1 | 1 |
| unsigned char | 1 | 1 | 1 | 1 | 1 | 1 |
| wchar\_t | 4 | 4 | 4 | 2 | 2 | 2 |
| short | 2 | 2 | 2 | 2 | 2 | 2 |
| unsigned short | 2 | 2 | 2 | 2 | 2 | 2 |
| int | 4 | 4 | 4 | 4 | 4 | 4 |
| unsigned int | 4 | 4 | 4 | 4 | 4 | 4 |
| long | 8 | 8 | 8 | 4 | 4 | 4 |
| unsigned long | 8 | 8 | 8 | 4 | 4 | 4 |
| long long | 8 | 8 | 8 | 8 | 8 | 8 |
| unsigned long long | 8 | 8 | 8 | 8 | 8 | 8 |
| float | 4 | 4 | 4 | 4 | 4 | 4 |
| double | 8 | 8 | 8 | 8 | 8 | 8 |
| long double | 16 | 16 | 16 | 16 | 8 | 8 |
| size\_t | 8 | 8 | 8 | 8 | 8 | 4 |
| ptrdiff\_t | 8 | 8 | 8 | 8 | 8 | 4 |
| void\* | 8 | 8 | 8 | 8 | 8 | 4 |
| char\* | 8 | 8 | 8 | 8 | 8 | 4 |
| int\* | 8 | 8 | 8 | 8 | 8 | 4 |
| unsigned int\* | 8 | 8 | 8 | 8 | 8 | 4 |

**Задание 2**

Выполните на платформах из таблицы Л2.1 задание Л1.з5. Убедитесь, что на всех платформах на основе x86 и x86-64 порядок байтов прямой



#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);

}

**GNU/Linux 64**

**GCC (x86-64 gcc 11.2):**

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

**clang (x86-64 clang 13.0.1):**

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

**ICC (x86-64 icc 2021.5.0):**

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

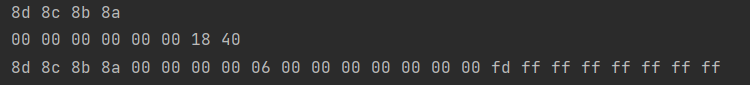
**MS Windows 64**

замена строки:

до: long mas[] = {0x8A8B8C8D, 6, -3};

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

**GCC (MinGW):**

****

**Microsoft (x64):**

****

**Microsoft (x86):**

****

**Итог:** действительно порядок байтов на всех платформах прямой.

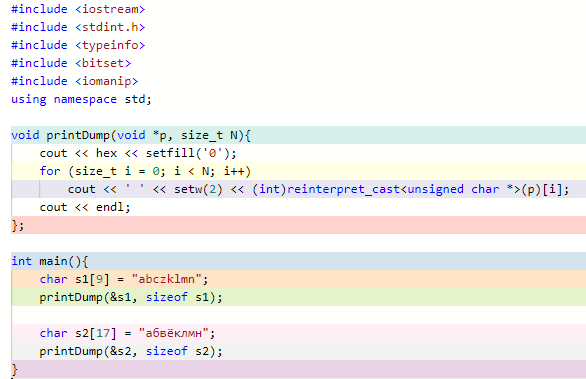
**Задание 3**

С помощью функции 𝑝𝑟𝑖𝑛𝑡𝐷𝑢𝑚𝑝() задания Л1.з5 определите и выпишите в отчёт, как хранятся в памяти на платформах из таблицы Л2.1:

– строки "abczklmn" и "абвёклмн" из char; при выборе количества отображаемых байтов 𝑁 учитывайте всю длину строки (включая завершающий нулевой символ), а не только видимые буквы;

– «широкие» строки L"abczklmn" и L"абвёклмн" из wxhar\_t; при выборе 𝑁 учитывайте всю длину строки. Результаты оформите в отчёте в виде таблицы.

**Char:**

****

**#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(){**

**char s1[9] = "abczklmn";**

**printDump(&s1, sizeof s1);**

**char s2[17] = "абвёклмн";**

**printDump(&s2, sizeof s2);**

**}**

**GNU/Linux 64**

**GCC (x86-64 gcc 11.2):**

61 62 63 7a 6b 6c 6d 6e 00

d0 b0 d0 b1 d0 b2 d1 91 d0 ba d0 bb d0 bc d0 bd 00

**clang (x86-64 clang 13.0.1):**

61 62 63 7a 6b 6c 6d 6e 00

d0 b0 d0 b1 d0 b2 d1 91 d0 ba d0 bb d0 bc d0 bd 00

**ICC (x86-64 icc 2021.5.0):**

61 62 63 7a 6b 6c 6d 6e 00

d0 b0 d0 b1 d0 b2 d1 91 d0 ba d0 bb d0 bc d0 bd 00

**MS Windows 64**

**GCC (MinGW):**

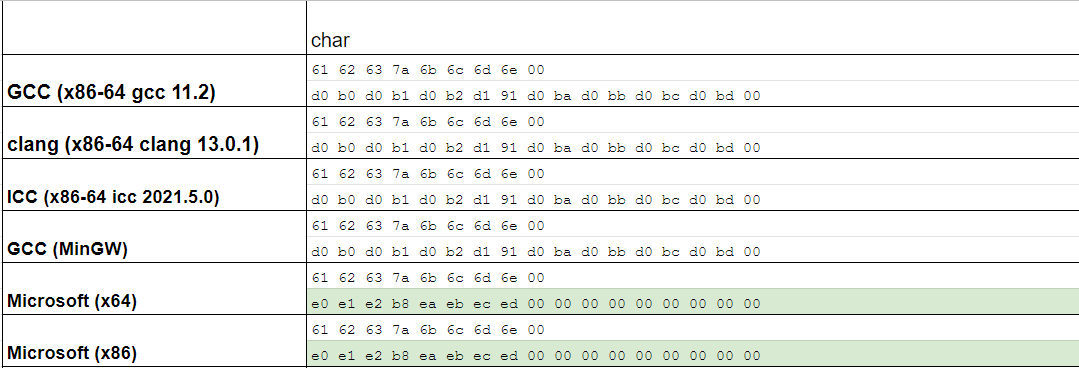


**Microsoft (x64):**

****

**Microsoft (x86):**

****



**Wchar\_t:**

****

**#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(){**

**wchar\_t \*s1 = L"abczklmn";**

**printDump(s1, 36);**

**wchar\_t \*s2 = L"абвёклмн";**

**printDump(s2, 36);**

**}**

**GNU/Linux 64**

**GCC (x86-64 gcc 11.2):**

61 00 00 00 62 00 00 00 63 00 00 00 7a 00 00 00 6b 00 00 00 6c 00 00 00 6d 00 00 00 6e 00 00 00 00 00 00 00

30 04 00 00 31 04 00 00 32 04 00 00 51 04 00 00 3a 04 00 00 3b 04 00 00 3c 04 00 00 3d 04 00 00 00 00 00 00

**clang (x86-64 clang 13.0.1):**

61 00 00 00 62 00 00 00 63 00 00 00 7a 00 00 00 6b 00 00 00 6c 00 00 00 6d 00 00 00 6e 00 00 00 00 00 00 00

30 04 00 00 31 04 00 00 32 04 00 00 51 04 00 00 3a 04 00 00 3b 04 00 00 3c 04 00 00 3d 04 00 00 00 00 00 00

**ICC (x86-64 icc 2021.5.0):**

61 00 00 00 62 00 00 00 63 00 00 00 7a 00 00 00 6b 00 00 00 6c 00 00 00 6d 00 00 00 6e 00 00 00 00 00 00 00

**invalid multibyte character sequence**

**MS Windows 64**

Замена:

**до: printDump(s1, 36);**

**после: printDump(s1, 18);**

**до: printDump(s2, 36);**

**после printDump(s2, 18);**

**GCC (MinGW):**

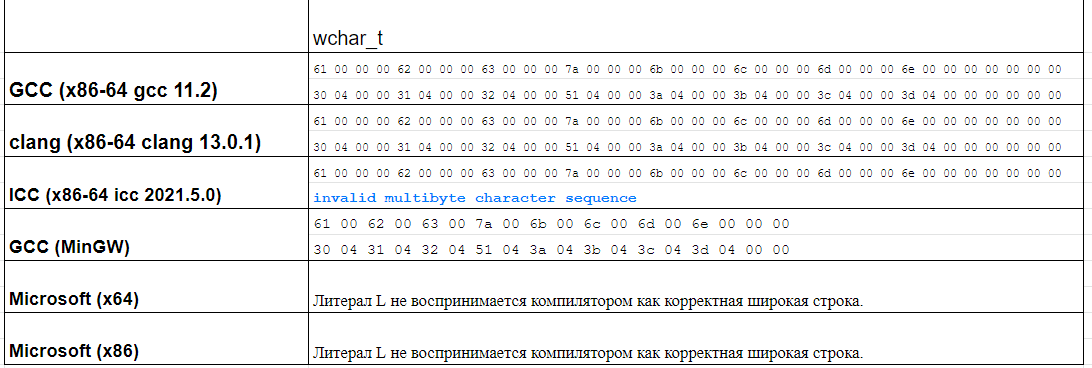
****

**Microsoft (x64):**

Литерал L не воспринимается компилятором как корректная широкая строка.

**Microsoft (x86):**

Литерал L не воспринимается компилятором как корректная широкая строка.



**Вопросы**

1. Как представляется символьная информация в компьютере в кодах ASCII, расширениях ASCII и различных кодировках Unicode?

Символьная информация в компьютере в кодах ASCII, расширениях ASCII и различных кодировках Unicode представляется в виде последовательности бит (байт). В кодировке ASCII длина кода каждого символа 8 бит (1 байт). Для Unicode -переменное **количество** **байт** (от 1 до 4).

2. Как хранятся русские буквы в «классических» и «широких» строках?

Для Linux:

В классических - по 2 байта на символ. В широких - по 4 байта.

Для Windows:

В классических - по 2 байта на символ. В широких - по 2 байта.

3. Как строковые функции libc (stdlib) определяют конец строки?

В языке C строки являются последовательностями символов, оканчивающимися символом '\0', также называемым «nul» (символ с кодом ноль)

4. Сколько символов (для узких строк — узких символов char, для широких — wchar\_t) необходимо для представления строки из пяти латинских букв? Цифр? Русских букв? Зависит ли ответ от платформы?

Ответ зависит от платформы.

Для узких строк (GNU/Linux 64):

Латинские буквы: каждый символ - 1 байт, каждый байт представим в виде двух шестнадцатеричных букв, то есть потребуется 5\*2=10 символов.

Русские буквы: каждый символ - 2 байта, каждый байт представим в виде двух шестнадцатеричных букв, то есть потребуется 5\*2\*2=20 символов.

Цифры: аналогично латинским буквам - 10 символов.

Для узких строк (GNU/Linux 64):

Латинские буквы: каждый символ - 4 байта, каждый байт представим в виде двух шестнадцатеричных букв, то есть потребуется 5\*4\*2=40 символов.

Русские буквы: аналогично латинским буквам - 40 символов.

Цифры: аналогично латинским буквам - 40 символов.