

Национальный исследовательский университет «Московский институт  
электронной техники»

# **Лабораторная работа №1**

**По «Архитектуре вычислительных систем»**

Выполнили: Зиновьева Е.

Ткаченко В.

*Зеленоград*

*2021*

**Задание 1.** Изучите, как интерпретируется одна и та же область памяти, если она рассматривается как знаковое или беззнаковое число, а также как одно и то же число записывается в различных системах счисления.

Необходимо сравнить:

- а) беззнаковую интерпретацию переменной в шестнадцатеричной форме;
- б) беззнаковую интерпретацию в десятичной форме;
- в) знаковую интерпретацию в десятичной форме.

```
#include <iostream>
#include <bitset>
#include <cmath>

void reckoning(int No) {
    std::cout << "Беззнаковая в 16-ричной форме: " << std::hex << (unsigned short)No << std::endl;
    std::cout << "Беззнаковая в 10-ричной форме: " << std::dec << (unsigned short)No << std::endl;
    std::cout << "Знаковая в 10-ричной форме: " << std::dec << (short)No << std::endl;
    std::cout << "2-чной форме: " << std::bitset<16>(No) << std::endl;
}

int main() {
    int x = pow(2, 15);

    std::cout << "x = -2^15:" << std::endl;
    reckoning(x);

    int y = pow(2, 15);

    std::cout << "y = 2^15:" << std::endl;
    reckoning(y);

    return 0;
}
```

Compiler Explorer interface showing the execution of the provided C++ code. The code calculates  $2^{15}$  and prints its representation in various formats. The output shows that for the same memory value (8000 in hex), the unsigned short interpretation is 8000 and the signed short interpretation is -32768.

Program returned: 0

Program stdout

x = -2^15:

Беззнаковая в 16-ричной форме: 8000  
Беззнаковая в 10-ричной форме: 32768  
Знаковая в 10-ричной форме: -32768  
2-чной форме: 1000000000000000

y = 2^15:

Беззнаковая в 16-ричной форме: 8000  
Беззнаковая в 10-ричной форме: 32768  
Знаковая в 10-ричной форме: -32768  
2-чной форме: 1000000000000000

**Задание 2.** Найдите и выпишите в отчёт минимальное и максимальное 16-битное число со знаком и без знака в формах представления (а), (б), (в) и в двоичной форме (4 числа, каждое из которых представлено в 4 формах).

```
#include <iostream>
#include "stdint.h"
#include "math.h"
#include <bitset>
#include <limits>

void reckoning(int No) {
    std::cout << "Беззнаковая в 16-ричной форме: " << std::hex << (unsigned short)No << std::endl;
    std::cout << "Беззнаковая в 10-ричной форме: " << std::dec << (unsigned short)No << std::endl;
    std::cout << "Знаковая в 10-ричной форме: " << std::dec << (short)No << std::endl;
    std::cout << "2-чной форме: " << std::bitset<16>(No) << std::endl;
}

int main(){
    int16_t min = std::numeric_limits<int16_t>::min();
    int16_t max = std::numeric_limits<int16_t>::max();
    uint16_t u_min = std::numeric_limits<uint16_t>::min();
    uint16_t u_max = std::numeric_limits<uint16_t>::max();
    std::cout << "Для максимального 16-битного значения со знаком" << std::endl;
    reckoning(max);
    std::cout << "Для минимального 16-битного значения со знаком" << std::endl;
    reckoning(min);
    std::cout << "Для максимального 16-битного беззнакового значения" << std::endl;
    reckoning(u_max);
    std::cout << "Для минимального 16-битного беззнакового значения" << std::endl;
    reckoning(u_min);

    return 0;
}
```

The screenshot shows the Compiler Explorer interface with the C++ code from the previous block. The output window displays the following results:

```
Для максимального 16-битного значения со знаком
Беззнаковая в 16-ричной форме: 7fff
Беззнаковая в 10-ричной форме: 32767
Знаковая в 10-ричной форме: 32767
2-чной форме: 0111111111111111

Для минимального 16-битного значения со знаком
Беззнаковая в 16-ричной форме: 8000
Беззнаковая в 10-ричной форме: 32768
Знаковая в 10-ричной форме: -32768
2-чной форме: 1000000000000000

Для максимального 16-битного беззнакового значения
Беззнаковая в 16-ричной форме: ffff
Беззнаковая в 10-ричной форме: 65535
Знаковая в 10-ричной форме: -1
2-чной форме: 1111111111111111

Для минимального 16-битного беззнакового значения
Беззнаковая в 16-ричной форме: 0
Беззнаковая в 10-ричной форме: 0
Знаковая в 10-ричной форме: 0
2-чной форме: 0000000000000000
```

**Задание 3.** Разработайте программу на языке C++, выполняющую над беззнаковыми шестнадцатитривиными целыми числами следующие поразрядные операции (результат должен печататься в десятичной и шестнадцатеричной формах):

бинарные

$x \wedge y$  (конъюнкция),

$x \vee y$  (дизъюнкция),

$x \oplus y$  (сложение по модулю два);

унарные

$\neg x$  (отрицание),

$\text{neg}(x)$  (дополнение до двух,  $x + \text{neg}(x) = 2^{\text{разрядность } x}$ );

$x \ll y$  (логический сдвиг влево),

$x \gg y$  (логический сдвиг вправо).

```
#include <iostream>
#include <cmath>

int main() {
    uint16_t x = 0x9211;
    uint16_t y = 0x0004;

    std::cout << "X_16 = " << std::hex << x;
    std::cout << " X_10 = " << std::dec << x << std::endl;

    std::cout << "y_16 = " << std::hex << y;
    std::cout << " y_10 = " << std::dec << y << std::endl;

    int i = 0, n = x;
    while(n > 0)
    {
        n = n / 10; ++i;
    }

    //Бинарные
    std::cout << "Конъюнкция :\n10-чной: " << std::dec << (x & y) << std::endl;
    std::cout << "16-ричной: " << std::hex << (x & y) << std::endl << std::endl;

    std::cout << "Дизъюнкция :\n10-чной: " << std::dec << (x | y) << std::endl;
    std::cout << "16-ричной: " << std::hex << (x | y) << std::endl << std::endl;

    std::cout << "Сложение по модулю 2 :\n10-чной: " << std::dec << (x ^ y) << std::endl;
    std::cout << "16-ричной: " << std::hex << (x ^ y) << std::endl << std::endl;

    //Унарные
    std::cout << "Отрицание :\n10-чной: " << std::dec << ~x << std::endl;
    std::cout << "16-ричной: " << std::hex << ~x << std::endl << std::endl;

    std::cout << "Дополнение до двух:\n10-чной: " << std::dec << (x + pow(2, n)) << std::endl;
    std::cout << "16-ричной: " << std::hex << (x + pow(2, n)) << std::endl << std::endl;

    std::cout << "Сдвиг влево:\n10-чной: " << std::dec << (x << y) << std::endl;
    std::cout << "16-ричной: " << std::hex << (x << y) << std::endl << std::endl;

    std::cout << "Сдвиг вправо:\n10-чной: " << std::dec << (x >> y) << std::endl;
    std::cout << "16-ричной: " << std::hex << (x >> y) << std::endl << std::endl;

    return 0;
}
```

```
}
```

The screenshot shows the Compiler Explorer interface with the following C++ code in the editor:

```
1 #include <iostream>
2 #include <cmath>
3
4 int main() {
5     uint16_t x = 0x9211;
6     uint16_t y = 0x0004;
7
8     int i = 0, n=x ; //Находим разрядность x
9     while(n>0)
10    {
11        n = n / 10; ++i;
12    }
13
14    //Бинарные
15    std::cout << "Конъюнкция : \n10-чной: " << std::dec<< (x&y);
16    std::cout << " 16-ричной: " << std::hex << (x&y) << std::endl <<
17
18    std::cout << "Дизъюнкция : \n10-чной: " << std::dec<< (x|y);
19    std::cout << " 16-ричной: " << std::hex << (x|y) << std::endl <<
20
21    std::cout << "Сложение по модулю 2 : \n10-чной: " << std::dec<< (x
22    std::cout << " 16-ричной: " << std::hex << (x^y) << std::endl <<
23
24    //Унарные
25    std::cout << "Отрицание : \n10-чной: " << std::dec<< ~x;
26    std::cout << " 16-ричной: " << std::hex << ~x << std::endl << std
27
28    std::cout << "Дополнение до двух: \n10-чной: " << std::dec<< (x+po
29    std::cout << " 16-ричной: " << std::hex << (x+pow(2,n)) << std::e
```

The output on the right shows the results of these operations:

Конъюнкция :  
10-чной: 0 16-ричной: 0

Дизъюнкция :  
10-чной: 37397 16-ричной: 9215

Сложение по модулю 2 :  
10-чной: 37397 16-ричной: 9215

Отрицание :  
10-чной: -37394 16-ричной: ffff6dee

Дополнение до двух:  
10-чной: 37394 16-ричной: 37394

Сдвиг влево:  
10-чной: 598288 16-ричной: 92110

Сдвиг вправо:  
10-чной: 2337 16-ричной: 921

**Задание 4.** Измените в программе из задания 3 тип переменных на знаковый. Объясните изменение (или неизменность) результата

The screenshot shows the Compiler Explorer interface with the modified C++ code in the editor:

```
1 #include <iostream>
2 #include <cmath>
3
4 int main() {
5     int16_t x = 0x9211;
6     int16_t y = 0x0004;
7
8     int i = 0, n=x ; //Находим разрядность x
9     while(n>0)
10    {
11        n = n / 10; ++i;
12    }
13
14    //Бинарные
15    std::cout << "Конъюнкция : \n10-чной: " << std::dec<< (x&y);
16    std::cout << " 16-ричной: " << std::hex << (x&y) << std::endl <<
17
18    std::cout << "Дизъюнкция : \n10-чной: " << std::dec<< (x|y);
19    std::cout << " 16-ричной: " << std::hex << (x|y) << std::endl <<
20
21    std::cout << "Сложение по модулю 2 : \n10-чной: " << std::dec<< (x
22    std::cout << " 16-ричной: " << std::hex << (x^y) << std::endl <<
23
24    //Унарные
25    std::cout << "Отрицание : \n10-чной: " << std::dec<< ~x;
26    std::cout << " 16-ричной: " << std::hex << ~x << std::endl << std
27
28    std::cout << "Дополнение до двух: \n10-чной: " << std::dec<< (x+po
29    std::cout << " 16-ричной: " << std::hex << (x+pow(2,n)) << std::e
```

The output on the right shows the results of these operations with signed integers:

Конъюнкция :  
10-чной: 0 16-ричной: 0

Дизъюнкция :  
10-чной: -28139 16-ричной: ffff9215

Сложение по модулю 2 :  
10-чной: -28139 16-ричной: ffff9215

Отрицание :  
10-чной: 28142 16-ричной: 6dee

Дополнение до двух:  
10-чной: -28143 16-ричной: -28143

Сдвиг влево:  
10-чной: -450288 16-ричной: fff92110

Сдвиг вправо:  
10-чной: -1759 16-ричной: fffff921

**Задание 5.** Бонус (+1 балл). Разработайте программу на языке C++ (или дополните программу из задания 3), которая расширяет шестнадцатитрибитное представление числа  $x$  до тридцатидвухбитного, рассматривая числа как

- знаковые (signed);
- беззнаковые (unsigned).

```
#include <iostream>
#include <cmath>

template<typename T>
void print(T x) {
    std::cout << "10-чная: " << std::hex << x;
    std::cout << " 2-чная: " << std::dec << x << std::endl;
}

int main() {
    int16_t x = 0x8001;
    int32_t x_change = x;

    std::cout << "Шестнадцатитрибитное представление (знаковое)" << std::endl;
    print(x);
    std::cout << "Тридцатидвухбитное представление (знаковое)" << std::endl;
    print(x_change);
    std::cout << std::endl;

    uint16_t y = 0x8001;
    uint32_t y_change = y;

    std::cout << "Шестнадцатитрибитное представление (беззнаковое)" << std::endl;
    print(y);

    std::cout << "Тридцатидвухбитное представление (беззнаковое)" << std::endl;
    print(y_change);

    return 0;
}
```

The screenshot shows the Compiler Explorer web interface. The code editor on the left contains the C++ program. The right panel shows the compiler output, which includes the program's return value (0) and its stdout output. The output displays the hexadecimal and decimal representations of the 16-bit and 32-bit signed and unsigned integers.

Compiler Explorer interface showing the C++ code and its execution output.

Code:

```
#include <iostream>
#include <cmath>

template<typename T>
void print(T x) {
    std::cout << "10-чная: " << std::hex << x;
    std::cout << " 2-чная: " << std::dec << x << std::endl;
}

int main() {
    int16_t x = 0x8001;
    int32_t x_change = x;

    std::cout << "Шестнадцатитрибитное представление (знаковое)" << std::endl;
    print(x);
    std::cout << "Тридцатидвухбитное представление (знаковое)" << std::endl;
    print(x_change);
    std::cout << std::endl;

    uint16_t y = 0x8001;
    uint32_t y_change = y;

    std::cout << "Шестнадцатитрибитное представление (беззнаковое)" << std::endl;
    print(y);

    std::cout << "Тридцатидвухбитное представление (беззнаковое)" << std::endl;
    print(y_change);

    return 0;
}
```

Output:

```
Program returned: 0
Program stdout
Шестнадцатитрибитное представление (знаковое)
10-чная: 8001 2-чная: -32767
Тридцатидвухбитное представление (знаковое)
10-чная: ffff8001 2-чная: -32767

Шестнадцатитрибитное представление (беззнаковое)
10-чная: 8001 2-чная: 32769
Тридцатидвухбитное представление (беззнаковое)
10-чная: 8001 2-чная: 32769
```

**Задание 6.** Определите и выпишите в отчёт, как хранятся в памяти компьютера: – целое число 0x12345678; по результату исследования определите порядок следования байтов в словах для вашего процессора:

- Little-Endian (от младшего к старшему, порядок Intel);
  - Big-Endian (от старшего к младшему, порядок Motorola);
- 320 Приложение А. Лабораторный практикум GNU Assembler
- строки "abcd" и "абвг" (массив из char);
  - «широкие» строки L"abcd" и L"абвг" (массив из wchar\_t).

```
#include <iostream>
#include <bitset>

int main()
{
    int32_t x = 0x12345678;;

    std::cout << "0x12345678" << std::endl;
    std::cout << "10-чная: " << std::dec << x << std::endl;
    std::cout << "2-чная: " << std::bitset<32>(x) << std::endl;

    char ENG[] = "abcd";
    char RUS[] = "абвг";

    std::cout << std::endl << "abcd:" << std::endl;
    for (int i = 0; i < 8; i++) {
        std::cout << (int)RUS[i] << " ";
    }

    std::cout << std::endl << "абвг" << std::endl;
    for (int i = 0; i < 4; i++) {
        std::cout << (int)ENG[i] << " ";
    }

    std::cout << std::endl;
    wchar_t expanded_ENG[] = L"abcd";
    wchar_t expanded_RUS[] = L"абвг";
    std::cout << std::endl << "abcd" << std::endl;
    for (int i = 0; i < 4; i++)
    {
        std::cout << (int)expanded_ENG[i] << " ";
    }
    std::cout << std::endl << "абвг" << std::endl;
    for (int i = 0; i < 4; i++)
    {
        std::cout << (int)expanded_RUS[i] << " ";
    }
    return 0;
}
```

The screenshot shows the Compiler Explorer interface. The left pane contains C++ source code for a program that demonstrates memory layout and bitsets. The right pane shows the compiler output, which includes the program's return value, standard output, and memory addresses for various variables.

```

1 #include <iostream>
2 #include <bitset>
3
4 int main()
5 {
6     int32_t x = 0x12345678;;
7
8     std::cout << "0x12345678" << std::endl;
9     std::cout << "10-чная: " << std::dec << x << std::endl;
10    std::cout << "2-чная: " << std::bitset<32>(x) << std::endl;
11
12    char ENG[] = "abcd";
13    char_RUS[] = "абвр";
14
15    std::cout << std::endl << "abcd:" << std::endl;
16    for (int i = 0; i < 8; i++) {
17        std::cout << (int)RUS[i] << " ";
18    }
19
20    std::cout << std::endl << "абвр:" << std::endl;
21    for (int i = 0; i < 4; i++) {
22        std::cout << (int)ENG[i] << " ";
23    }
24
25    std::cout << std::endl;
26    wchar_t expanded_ENG[] = L"abcd";
27    wchar_t expanded_RUS[] = L"абвр";
28    std::cout << std::endl << "abcd:" << std::endl;
29    for (int i = 0; i < 4; i++)
  
```

Compiler output:

```

Program returned: 0
Program stdout
0x12345678
10-чная: 305419896
2-чная: 00010010001101000101011001111000

abcd:
-48 -80 -48 -79 -48 -78 -48 -77
абвр
97 98 99 100

abcd
97 98 99 100
абвр
1072 1073 1074 1075
  
```

**Задание 7.** При помощи оператора `sizeof` выясните, сколько байтов занимают переменные следующих типов: `char`, `bool`, `wchar_t`, `short`, `int`, `long`, `long long`, `float`, `double`, `long double`, `size_t`, `ptrdiff_t`, `void*`. Результаты оформите в отчёте в виде таблицы, указывая для каждого типа его назначение.

Для выполнения единообразных действий над переменными различных типов используются макросы препроцессора C или шаблоны C++.

Проверьте, соответствуют ли размеры типов современному стандарту C++.

```

#include <iostream>
#include <string>

template<typename T>

void size_of(std::string type_name){
    std::cout << "Размер " << type_name << " - " << sizeof(T) << std::endl;
}

int main(){
    size_of<char>("char");
    size_of<bool>("bool");
    size_of<wchar_t>("wchar_t");
    size_of<short>("short");
    size_of<int>("int");
    size_of<long>("long");
    size_of<long long>("long long");
    size_of<float>("float");
    size_of<double>("double");
    size_of<long double>("long double");
    size_of<size_t>("size_t");
    size_of<ptrdiff_t>("ptrdiff_t");
    size_of<void*>("void*");

    return 0;
}
  
```



The screenshot shows the Compiler Explorer interface. The left pane contains C++ source code that defines a function to print the size of various types and a main function that calls this function for several types. The right pane shows the compiler output, which lists the size of each type in bytes, with Russian labels like 'Размер' (Size) and 'Размер' (Size) preceding the values.

```

1 #include <iostream>
2 #include <string>
3
4 template<typename T>
5
6 void size_of(std::string type_name){
7     std::cout << "Размер " << type_name << " - " << sizeof(T) << std::endl;
8 }
9
10 int main(){
11     size_of<char>("char");
12     size_of<bool>("bool");
13     size_of<wchar_t>("wchar_t");
14     size_of<short>("short");
15     size_of<int>("int");
16     size_of<long>("long");
17     size_of<long long>("long long");
18     size_of<float>("float");
19     size_of<double>("double");
20     size_of<long double>("long double");
21     size_of<size_t>("size_t");
22     size_of<ptrdiff_t>("ptrdiff_t");
23     size_of<void*>("void*");
24
25     return 0;
26 }

```

```

Program returned: 0
Program stdout
Размер char - 1
Размер bool - 1
Размер wchar_t - 4
Размер short - 2
Размер int - 4
Размер long - 8
Размер long long - 8
Размер float - 4
Размер double - 8
Размер long double - 16
Размер size_t - 8
Размер ptrdiff_t - 8
Размер void* - 8

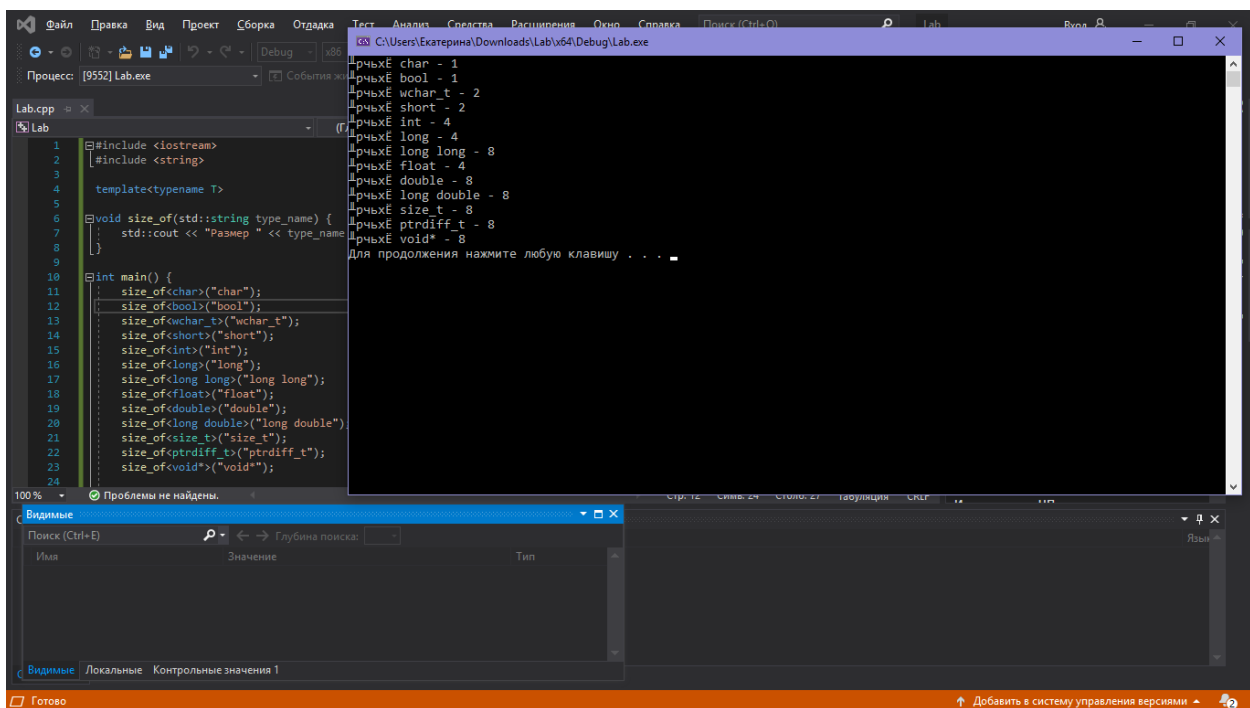
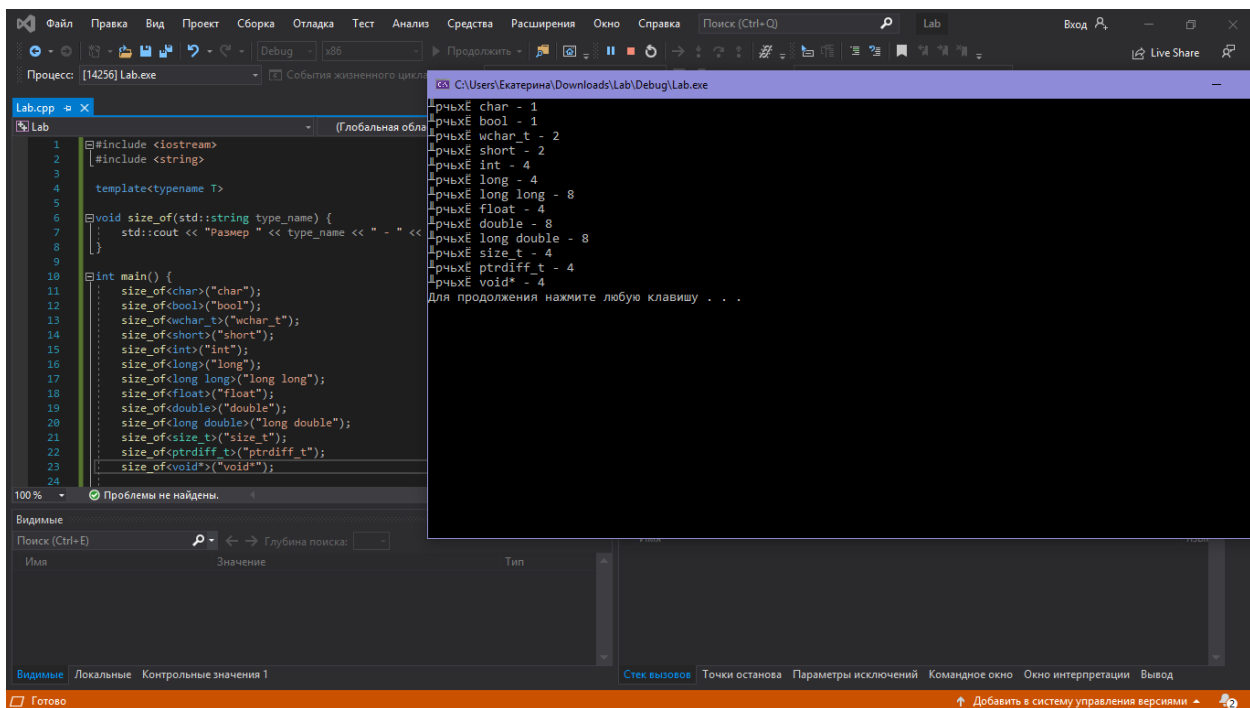
```

**Задание 8.** Запустите программу (программы) из заданий 6–7 на двух других платформах, доступных на ВЦ — 32- и 64-разрядной версиях Microsoft Windows и повторите измерения. Для каждого из заданий дополните таблицу результатами новых измерений. Платформы, для которых необходимо провести измерения:

- GNU/Linux Ubuntu, архитектура x86 (компилятор GCC, среда любая);
- 32-разрядная версия Microsoft Windows (ВЦ, среда Microsoft Visual Studio);
- 64-разрядная версия Microsoft Windows (терминал Skylab, среда Microsoft Visual Studio, 64-битная конфигурация).

Результаты однотипных измерений, выполненных на различных платформах, должны быть сгруппированы в таблицу или последовательно описаны в одном разделе.

Тип данных	Win64	Win32	Ubuntu x32	Ubuntu x64
Char	1	1	1	
Bool	1	1	1	
Wchar t	2	2	4	
Short	2	2	2	
Int	4	4	4	
Long	4	4	8	
Long long	8	8	8	
Float	4	4	4	
Double	8	8	8	
Long double	8	8	16	
Size t	8	4	8	
Pttdiff t	8	4	8	
Void*	8	4	8	



Compiler Explorer

godbolt.org

COMPILER EXPLORER

Add... More

Support diversity in C++ with #include <C++>

SponsorsPC-lintSilly Labs

ShareOtherPolicies

x86-64 gcc 10.2 (Editor #1, Compiler #1) C++

C++ source #1

x86-64 gcc 10.2 (Editor #1, Compiler #1) C++

x86-64 gcc 10.2

Compiler options...

Save/LoadAdd new... VimCpplnsightsQuick-benchC++

```
1 #include <iostream>
2 #include <string>
3
4 template<typename T>
5
6 void size_of(std::string type_name){
7     std::cout << "Размер " << type_name << " - " << sizeof(T) << std::endl;
8 }
9
10 int main(){
11     size_of<char>("char");
12     size_of<bool>("bool");
13     size_of<wchar_t>("wchar_t");
14     size_of<short>("short");
15     size_of<int>("int");
16     size_of<long>("long");
17     size_of<long long>("long long");
18     size_of<float>("float");
19     size_of<double>("double");
20     size_of<long double>("long double");
21     size_of<size_t>("size_t");
22     size_of<ptrdiff_t>("ptrdiff_t");
23     size_of<void*>("void*");
24
25     return 0;
26 }
```

Program returned: 0

Program stdout

Размер char - 1
Размер bool - 1
Размер wchar\_t - 4
Размер short - 2
Размер int - 4
Размер long - 8
Размер long long - 8
Размер float - 4
Размер double - 8
Размер long double - 16
Размер size\_t - 8
Размер ptrdiff\_t - 8
Размер void\* - 8

x86-64 gcc 10.2

1807ms

Read the new cookie policy

Compiler Explorer uses cookies and other related techs to serve you

Consent

Don't consent