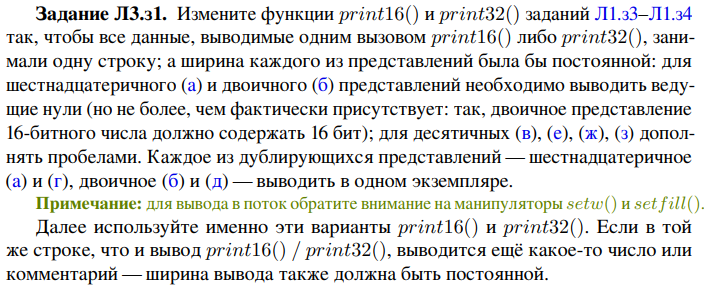
Лабораторная работа 3 (0011 = 3) Целочисленная арифметика и арифметика с плавающей запятой

****

void print16(void \*p)

{

cout<<setw(4)<<"u16:"<<hex<<setw(16)<<setfill('0')<<\*(reinterpret\_cast<unsigned short\*>(p))<<endl;

cout<<setw(4)<<setfill(' ')<<"u2:"<<setw(16)<<setfill('0')<< bitset<16>{\*(reinterpret\_cast<unsigned short\*>(p))}<<endl;

cout<<setw(4)<<"u10:"<<setw(16)<<setfill(' ')<< dec<<\*(reinterpret\_cast<unsigned short\*>(p))<<endl;

cout<<setw(4)<<"s10:"<<setw(16)<<setfill(' ')<< dec<< \*(reinterpret\_cast<short\*>(p))<<endl<<endl;

}

template<typename T>

void print32(T \*p)

{

cout<<setw(5)<<""<<setw(32)<<\*p<<endl;

cout<<setw(5)<<"16: "<<hex<<setw(32)<<setfill('0')<<\*(reinterpret\_cast<unsigned long\*>(p))<<setw(4)<<endl;

cout<<setfill(' ');

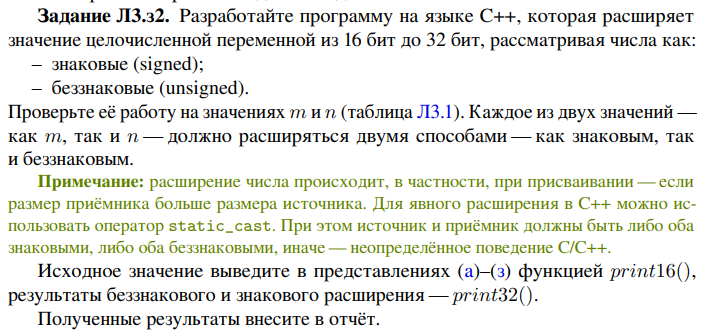
cout<<setw(5)<<"2: "<< setw(32)<<setfill('0')<<bitset<32>{\*(reinterpret\_cast<unsigned long\*>(p))}<<endl;

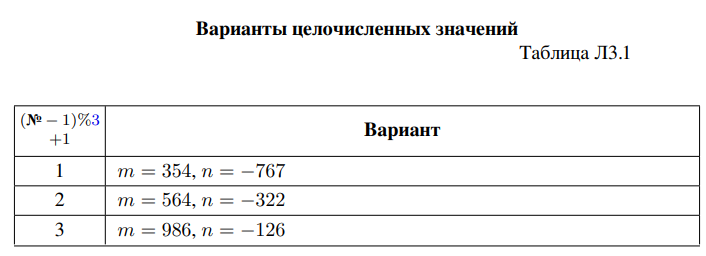
cout<<setfill(' ');

cout<<setw(5)<<"u10: "<<dec<<setw(32)<<\*(reinterpret\_cast<unsigned long\*>(p))<<endl;

cout<<setw(5)<<"s10: "<<dec<<setw(32)<< \*(reinterpret\_cast<long\*>(p))<<endl<<endl;

}





#include <iostream>

#include <cstdlib>

#include <array>

#include <bitset>

#include <iomanip>

using namespace std;

void print16(void \*p)

{

cout<<setw(4)<<"u16:"<<hex<<setw(16)<<setfill('0')<<\*(reinterpret\_cast<unsigned short\*>(p))<<endl;

cout<<setw(4)<<setfill(' ')<<"u2:"<<setw(16)<<setfill('0')<< bitset<16>{\*(reinterpret\_cast<unsigned short\*>(p))}<<endl;

cout<<setw(4)<<"u10:"<<setw(16)<<setfill(' ')<< dec<<\*(reinterpret\_cast<unsigned short\*>(p))<<endl;

cout<<setw(4)<<"s10:"<<setw(16)<<setfill(' ')<< dec<< \*(reinterpret\_cast<short\*>(p))<<endl<<endl;

}

template<typename T>

void print32(T \*p)

{

cout<<setw(5)<<""<<setw(32)<<\*p<<endl;

cout<<setw(5)<<"16: "<<hex<<setw(32)<<setfill('0')<<\*(reinterpret\_cast<unsigned long\*>(p))<<setw(4)<<endl;

cout<<setfill(' ');

cout<<setw(5)<<"2: "<< setw(32)<<setfill('0')<<bitset<32>{\*(reinterpret\_cast<unsigned long\*>(p))}<<endl;

cout<<setfill(' ');

cout<<setw(5)<<"u10: "<<dec<<setw(32)<<\*(reinterpret\_cast<unsigned long\*>(p))<<endl;

cout<<setw(5)<<"s10: "<<dec<<setw(32)<< \*(reinterpret\_cast<long\*>(p))<<endl<<endl;

}

template<typename T>

void expand(T p)

{

unsigned long unsigned16;

long signed16;

unsigned16 = p;

signed16 = p;

cout<<"Value: "<<p<<endl;

cout<<"Unsigned expand"<<endl;

print32(&unsigned16);

cout<<"Signed expand"<<endl;

print32(&signed16);

}

int main()

{

unsigned int m = 564;

int n = -322;

print16(&m);

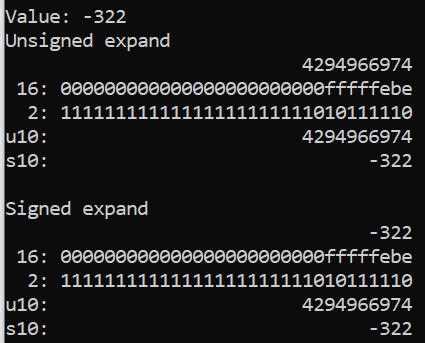
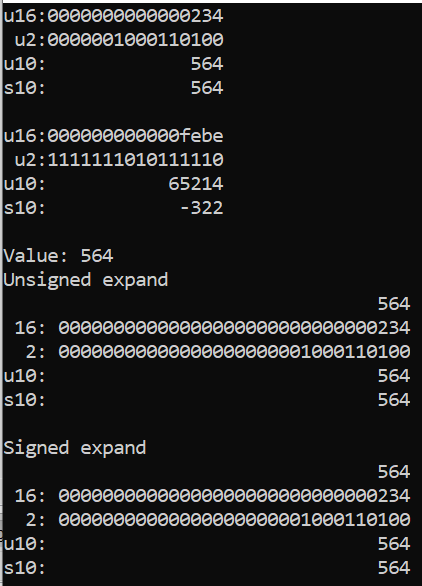
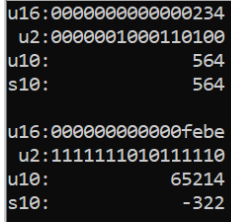
print16(&n);

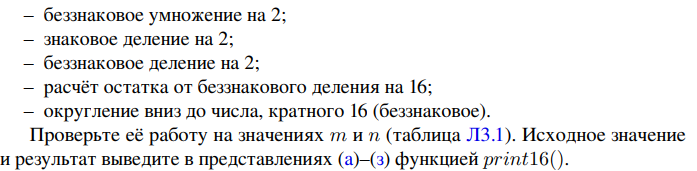
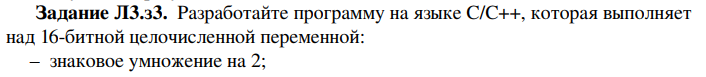
expand(m);

expand(n);

return 0;

}





|  | **m=564** | **n=-322** |
| --- | --- | --- |
| **Исходное значение** |  |  |
| **Знаковое умножение на 2** |  |  |
| **Беззнаковое умножение на 2** |  |  |
| **Знаковое деление на 2** |  |  |
| **Беззнаковое деление на 2** |  |  |
| **Остаток от беззнакового деления на 16** |  |  |
| **Округление вниз до числа, кратного 16(беззнаковое)** |  |  |

#include <iostream>

#include <cstdlib>

#include <array>

#include <bitset>

#include <windows.h>

#include <iomanip>

#include <stdio.h>

using namespace std;

void print16(void \*p)

{

cout<<setw(4)<<"u16:"<<hex<<setw(16)<<setfill('0')<<\*(reinterpret\_cast<unsigned short\*>(p))<<endl;

cout<<setw(4)<<setfill(' ')<<"u2:"<<setw(16)<<setfill('0')<< bitset<16>{\*(reinterpret\_cast<unsigned short\*>(p))}<<endl;

cout<<setw(4)<<"u10:"<<setw(16)<<setfill(' ')<< dec<<\*(reinterpret\_cast<unsigned short\*>(p))<<endl;

cout<<setw(4)<<"s10:"<<setw(16)<<setfill(' ')<< dec<< \*(reinterpret\_cast<short\*>(p))<<endl<<endl;

}

void ex3(short temp)

{

cout<<"for "<<temp<<":"<<endl<<"signed 2 x"<<endl;

short r1=temp\*2;

print16(&r1);

unsigned short r6=temp\*2u;

cout<<"unsigned 2 x"<<endl;

print16(&r6);

short r2=temp/2;

cout<<"signed div 2"<<endl;

print16(&r2);

unsigned short r3=temp/2u;

cout<<"unsigned div 2"<<endl;

print16(&r3);

unsigned short r4=temp%16u;

cout<<"rest after unsigned div 16"<<endl;

print16(&r4);

unsigned short r5=temp - (temp>-1 ? abs(temp)%16u : (16u-abs(temp)%16u));

cout<<"round down to unsigned 16k"<<endl;

print16(&r5);

}

int main()

{

short m=564;

short n=-322;

print16(&m);

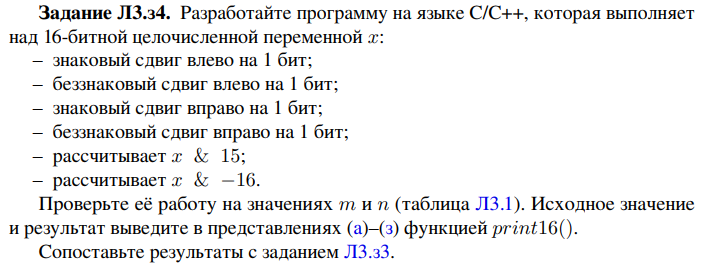
ex3(m);

print16(&n);

ex3(n);

return 0;

}



| Операция | m = 564 | n = -322 |
| --- | --- | --- |
| Исходное значение |  |  |
| Знаковый сдвиг влево на 1 бит |  |  |
| Беззнаковый сдвиг влево на 1 бит |  |  |
| Знаковый сдвиг вправо на 1 бит |  |  |
| Беззнаковый сдвиг вправо на 1 бит |  |  |
| х & 15 |  |  |
| x & -16 |  |  |

#include <iostream>

#include <cstdlib>

#include <array>

#include <bitset>

#include <iomanip>

using namespace std;

void print16(void \*p)

{

cout<<setw(4)<<"u16:"<<hex<<setw(16)<<setfill('0')<<\*(reinterpret\_cast<unsigned short\*>(p))<<endl;

cout<<setw(4)<<setfill(' ')<<"u2:"<<setw(16)<<setfill('0')<< bitset<16>{\*(reinterpret\_cast<unsigned short\*>(p))}<<endl;

cout<<setw(4)<<"u10:"<<setw(16)<<setfill(' ')<< dec<<\*(reinterpret\_cast<unsigned short\*>(p))<<endl;

cout<<setw(4)<<"s10:"<<setw(16)<<setfill(' ')<< dec<< \*(reinterpret\_cast<short\*>(p))<<endl<<endl;

}

template<typename T>

void print32(T \*p)

{

cout<<setw(5)<<""<<setw(32)<<\*p<<endl;

cout<<setw(5)<<"16: "<<hex<<setw(32)<<setfill('0')<<\*(reinterpret\_cast<unsigned long\*>(p))<<setw(4)<<endl;

cout<<setfill(' ');

cout<<setw(5)<<"2: "<< setw(32)<<setfill('0')<<bitset<32>{\*(reinterpret\_cast<unsigned long\*>(p))}<<endl;

cout<<setfill(' ');

cout<<setw(5)<<"u10: "<<dec<<setw(32)<<\*(reinterpret\_cast<unsigned long\*>(p))<<endl;

cout<<setw(5)<<"s10: "<<dec<<setw(32)<< \*(reinterpret\_cast<long\*>(p))<<endl<<endl;

}

template<typename T>

void shift(T p)

{

unsigned int unsigned\_shift = 1;

int signed\_shift = 1;

auto ans\_signed\_left = p<<signed\_shift;

auto ans\_unsigned\_left = p<<unsigned\_shift;

auto ans\_signed\_right = p>>signed\_shift;

auto ans\_unsigned\_right = p>>unsigned\_shift;

cout<<"Signed shift left"<<endl;

print32(&ans\_signed\_left);

cout<<"Unsigned shift left"<<endl;

print32(&ans\_unsigned\_left);

cout<<"Signed shift right"<<endl;

print32(&ans\_signed\_right);

cout<<"Unsigned shift right"<<endl;

print32(&ans\_unsigned\_right);

auto ans15 = p&15;

cout<<"p & 15"<<endl;

print32(&ans15);

auto ans16 = p&(-16);

cout<<"p & -16"<<endl;

print32(&ans16);

}

int main()

{

unsigned int m = 564;

int n = -322;

print16(&m);

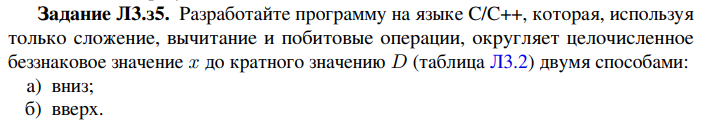
shift(m);

print16(&n);

shift(n);

return 0;

}



#include <iostream>

#include <iomanip>

using namespace std;

void round(unsigned short n, int d)

{

int temp=n;

int rest=n & (d-1); //поразрядная конъюнкция и вычитание

cout<<"value: "<<n<<endl;

while (rest!=0)

{

n=n-1; //вычитание

rest=n & (d-1); //поразрядная конъюнкция и вычитание

}

cout<<"down: "<<n<<endl;

rest=temp & (d-1); //поразрядная конъюнкция и вычитание

while (rest!=0)

{

temp=temp+1; //сложение

rest=temp & (d-1); //поразрядная конъюнкция и вычитание

}

if (temp<65535) cout<<"up: "<<temp<<endl;

else cout<<"up: "<<temp-65536<<endl; //вычитание

}

int main()

{ const int D=32;

round(97u,D);

round(1030u,D);

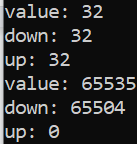
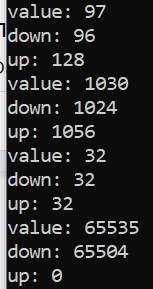
round(32,D);

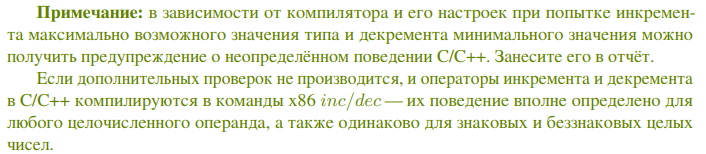
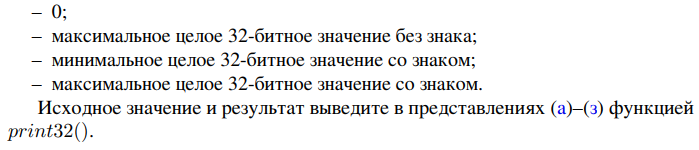
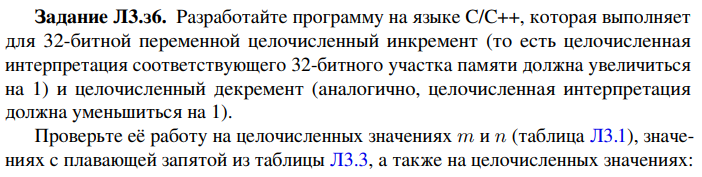
round(65535u,D);

return 0;

}

Проверка на каких-нибудь числах:





| Число | Исходное значение | Инкремент | Декремент |
| --- | --- | --- | --- |
| int m=564 |  |  |  |
| int n=-322 |  |  |  |
| float a=0 |  |  |  |
| float b=1 |  |  |  |
| float c=12233445 |  |  |  |
| float d=122334455 |  |  |  |
| int 0 |  |  |  |
| макс. цел. 32 бит без знака |  |  |  |
| мин. цел. 32 бит знак. |  |  |  |
| макс. цел. 32 бит знак. |  |  |  |

#include <iostream>

#include <iomanip>

#include <bitset>

using namespace std;

template<typename T>

void print32(T \*p)

{

cout<<setw(15)<<"Value: "<<setw(16)<<\*p<<endl;

cout<<setw(15)<<"Unsigned 16: "<<hex<<setw(16)<<setfill('0')<<\*(reinterpret\_cast<unsigned long\*>(p))<<setw(4)<<endl;

cout<<setfill(' ');

cout<<setw(15)<<"Unsigned 2: "<< setw(16)<<setfill('0')<<bitset<8>{\*(reinterpret\_cast<unsigned long\*>(p))}<<endl;

cout<<setfill(' ');

cout<<setw(15)<<"Unsigned 10: "<<dec<<setw(16)<<\*(reinterpret\_cast<unsigned long\*>(p))<<endl;

cout<<setw(15)<<"Signed 10: "<<dec<<setw(16)<< \*(reinterpret\_cast<long\*>(p))<<endl<<endl;

}

template<typename T>

void increment(T \*p)

{

\*p=\*p+1;

print32(p);

}

template<typename T>

void decrement(T \*p)

{

\*p=\*p-1;

print32(p);

}

int main()

{

int m=564;

int n=-322;

float a=0;

float b=1;

float c=12233445;

float d=122334455;

int n1=0;

unsigned int n2=4294967295;

int n3=-2147483648;

int n4=2147483647;

print32(&m);

print32(&n);

print32(&a);

print32(&b);

print32(&c);

print32(&d);

print32(&n1);

print32(&n2);

print32(&n3);

print32(&n4);

increment(&m);

increment(&n);

increment(&a);

increment(&b);

increment(&c);

increment(&d);

increment(&n1);

increment(&n2);

increment(&n3);

increment(&n4);

m=564;

n=-322;

a=0;

b=1;

c=12233445;

d=122334455;

n1=0;

n2=4294967295;

n3=-2147483648;

n4=2147483647;

decrement(&m);

decrement(&n);

decrement(&a);

decrement(&b);

decrement(&c);

decrement(&d);

decrement(&n1);

decrement(&n2);

decrement(&n3);

decrement(&n4);

return 0;

}