**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра Вычислительной техники**

отчет

**по лабораторной работе № 1**

**по дисциплине «Организация ЭВМ и систем»**

Тема: **«Исследование внутреннего представления различных форматов данных»**

|  |  |  |
| --- | --- | --- |
| Студенты |  |  |
| Преподаватель |  | Гречухин М.Н. |

Санкт-Петербург

2021

**Цель работы.**

1. Ознакомиться с внутренними форматами представления данных.
2. Вспомнить поразрядные логические операции в языке С++, указатели и операции над ними, структуры типа union и функции работы с файлами.

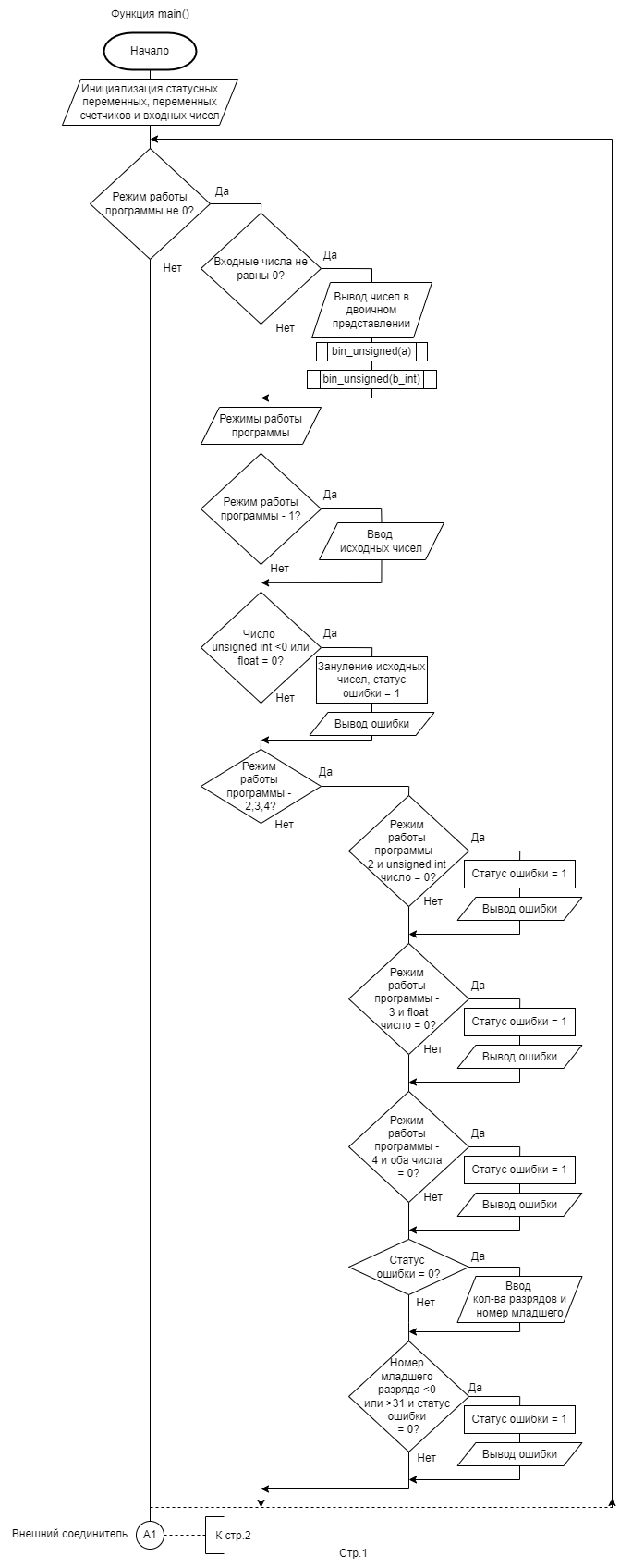
**Задание (вариант №13)**

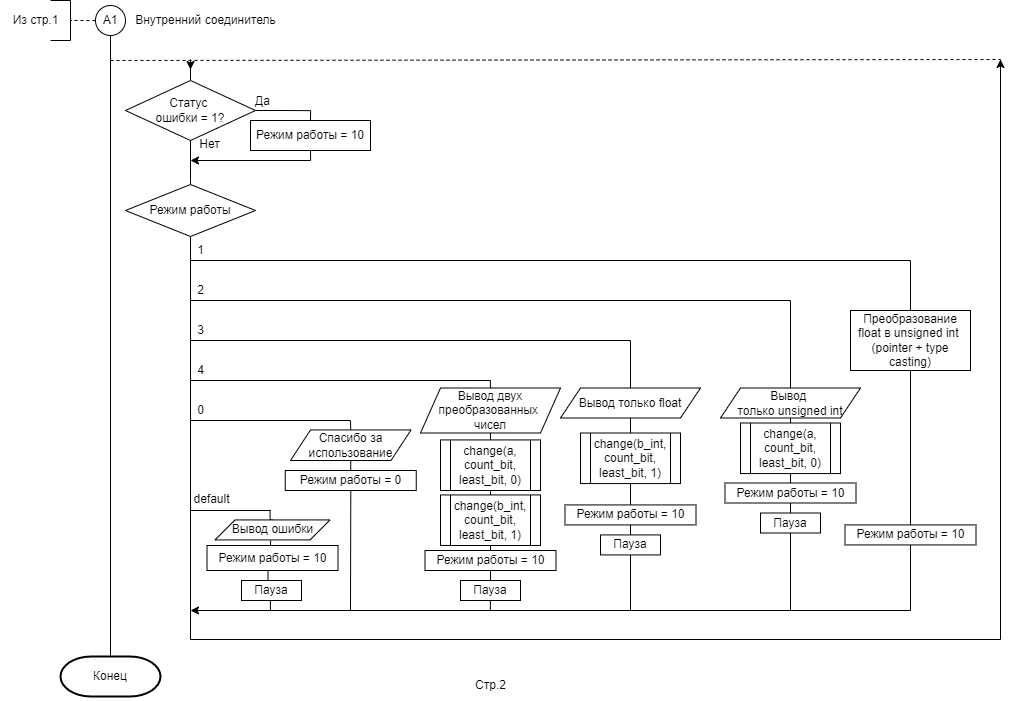
Выполнить зеркальную перестановку в группе рядом стоящих разрядов, количество которых и номер младшего разряда в группе задаются с клавиатуры.

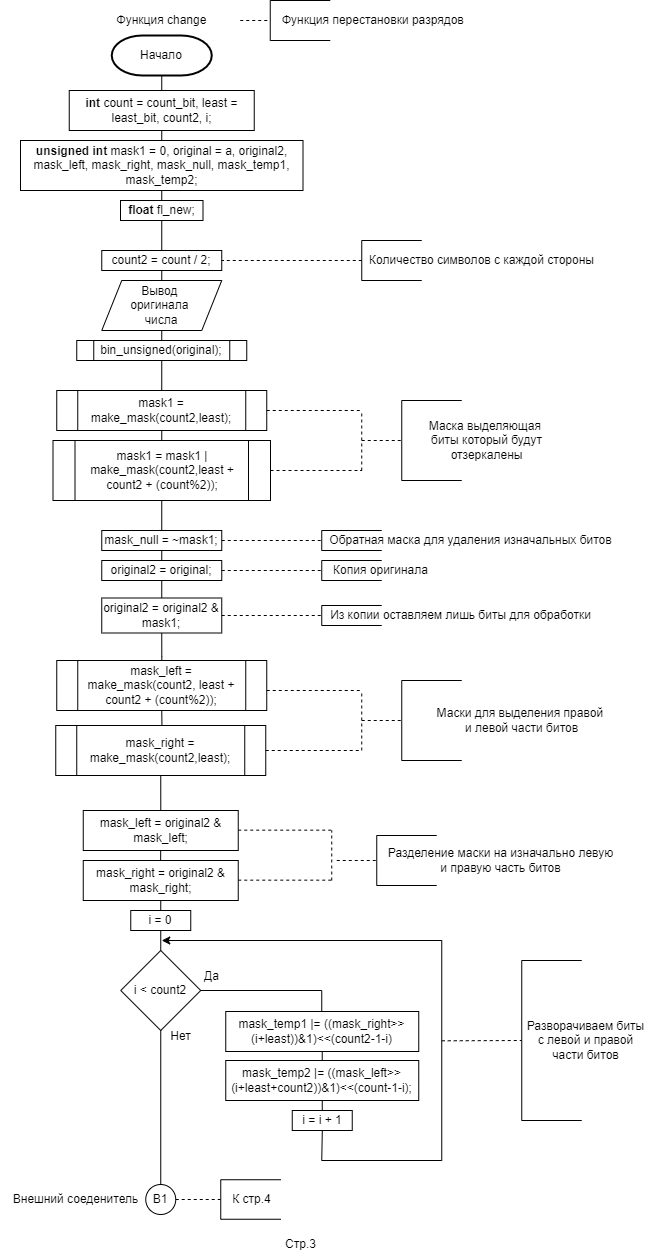
**Программная среда:**

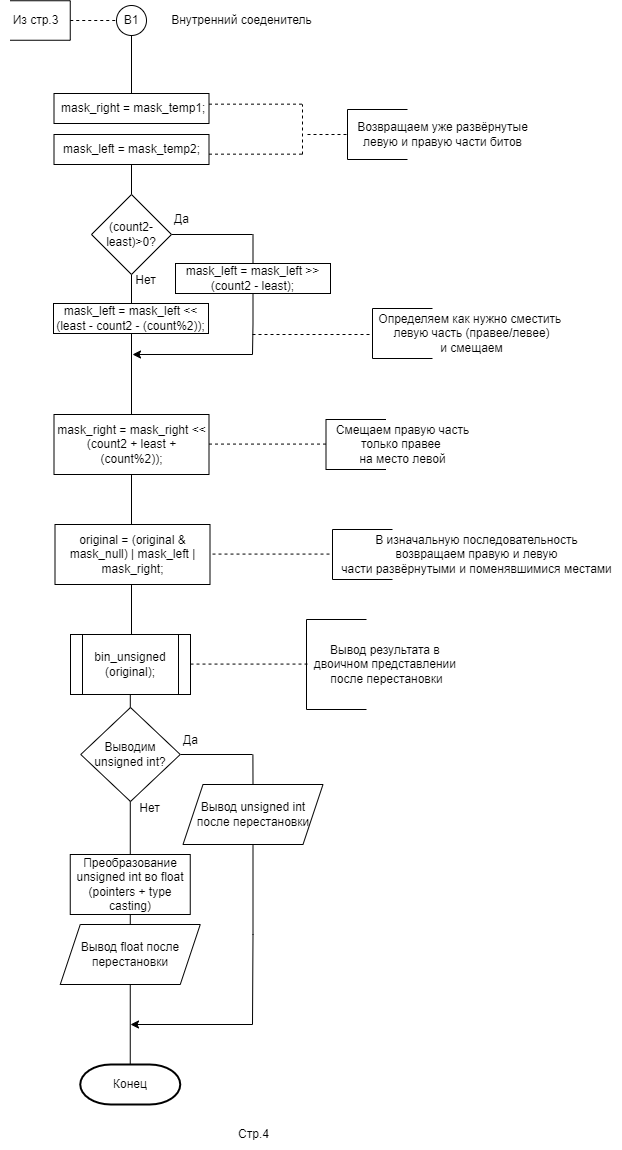
Вся программа написана на языке программирования С++, компилировалась в IDE Code Blocks, GNU GCC Compiler (without compiler flags).

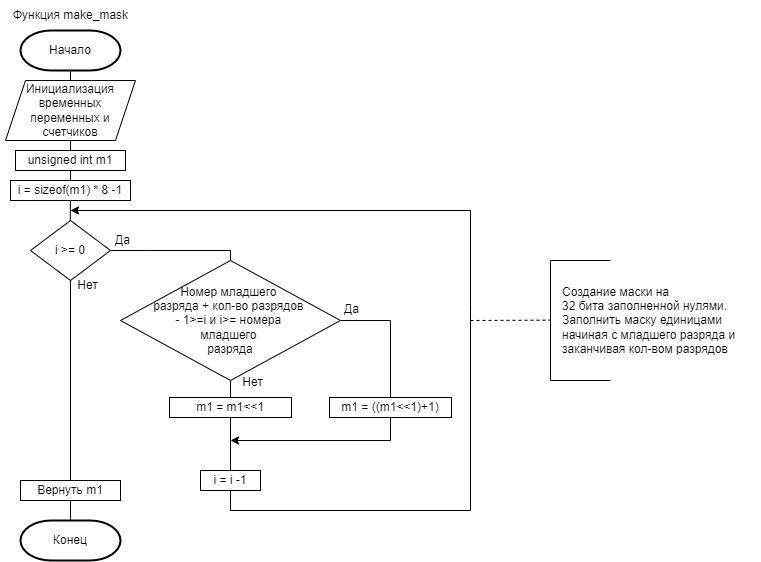
**Схемы алгоритмов**

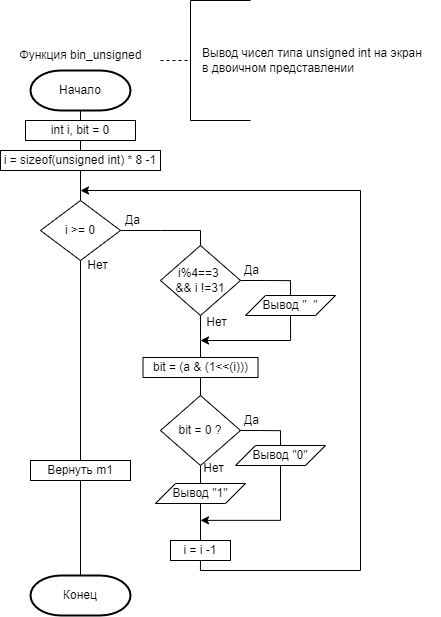


****

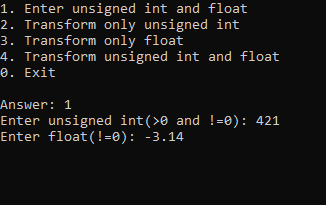
****



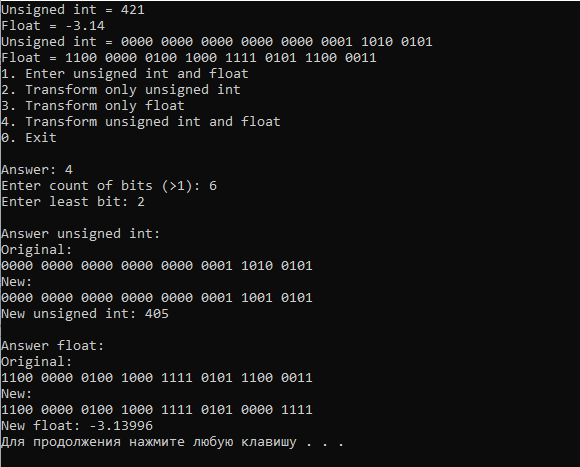




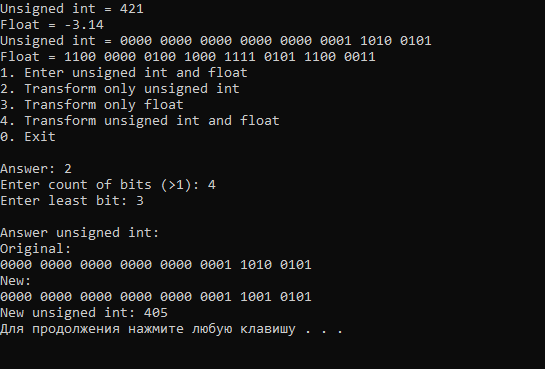
**Примеры работы программы**

****

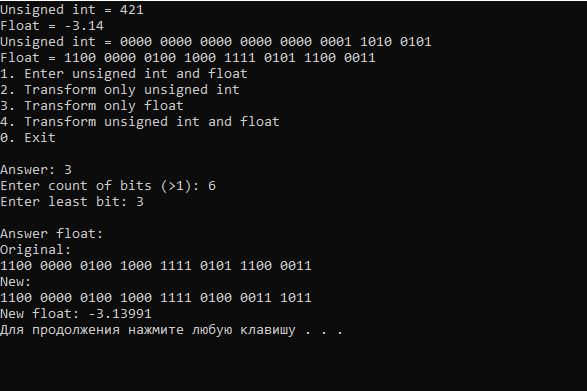
**Ввод исходных данных**

****

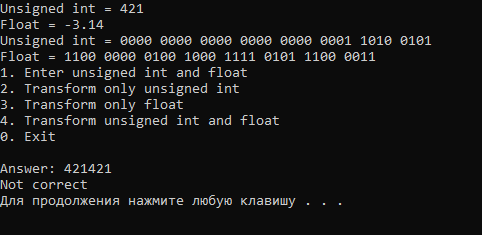
**Перестановка разрядов в обоих числах**

****

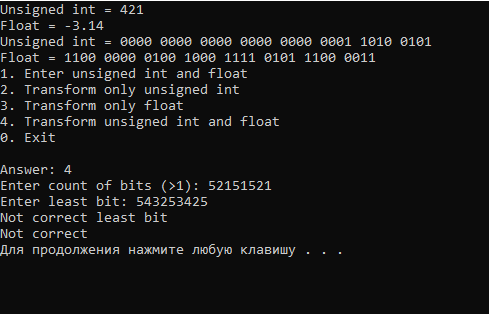
**Перестановка только в unsigned int**

****

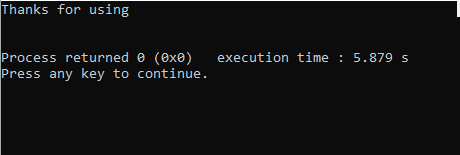
**Перестановка только во float**

****

**Ошибочный ввод**

****

**Ошибочный ввод**

****

**Конец программы**

**Текст программы**

#include <iostream>

#include <algorithm>

#include <string>

#include <random>

#include <iterator>

#include <ctime>

#include <Windows.h>

#define clear system("clear||@cls");

using namespace std;

void bin\_unsigned(unsigned int a){

int i = 0, bit = 0;

/\* Вывод переменной unsigned int в виде двоичного кода \*/

for(i=sizeof(a)\*8-1;i>=0;i--)

{

if (i%4==3 && i!=31) cout<<" ";

bit = (a & (1<<(i)));

if(bit == 0)

{

cout<<'0';

}

else

{

cout<<'1';

}

}

cout<< endl;

}

unsigned int make\_mask(int count\_bit, int least\_bit){

unsigned int m1;

int i, least = least\_bit, count = count\_bit;

/\* Создаём маску на 32 бита только из единиц и возвращаем её \*/

for(i=sizeof(m1)\*8-1;i>=0;i--)

{

if ((least + count - 1 >= i)&&(i >= least)){

m1 = ((m1<<1)+1);

}

else{

m1 = m1<<1;

}

}

return m1;

}

void change(unsigned int a, int count\_bit, int least\_bit, int F\_or\_I){

int count = count\_bit, least = least\_bit, count2, i;

unsigned int mask1 = 0, original = a, original2, mask\_left, mask\_right, mask\_null, mask\_temp1, mask\_temp2;

float fl\_new;

/\* Сколько символов надо зеркалить \*/

count2 = count / 2;

cout<<"Original:"<<endl;

bin\_unsigned(original);

/\* Маска из единиц для доставания нужных битов \*/

mask1 = make\_mask(count2,least);

mask1 = mask1 | make\_mask(count2,least + count2 + (count%2));

/\* Маска для удаления битов которые зеркаляться и копия оригинала \*/

mask\_null = ~mask1;

original2 = original;

/\* Вытаскиваем нужные биты \*/

original2 = original2 & mask1;

/\* Маски для разделения на левую и правую стороны \*/

mask\_left = make\_mask(count2, least + count2 + (count%2));

mask\_right = make\_mask(count2,least);

/\* Вытаскиваем левую и правую стороны, готовим временные переменные, где будем зеркалить каждую сторону \*/

mask\_left = original2 & mask\_left;

mask\_right = original2 & mask\_right;

mask\_temp1 = 0;

mask\_temp2 = 0;

/\* Сдвигаем в правый край биты, просматриваем первый и ставим его на новое место с другой стороны во временной переменной, повторяем столько, сколько бит с каждой стороны \*/

for(i = 0; i <= count2; i++){

mask\_temp1 |= ((mask\_right>>(i+least))&1)<<(count2-1-i);

mask\_temp2 |= ((mask\_left>>(i+least+count2))&1)<<(count-1-i);

}

/\* Возвращаем уже зеркаленные биты обратно в маски право/лево \*/

mask\_right = mask\_temp1;

mask\_left = mask\_temp2;

/\* Смотря в каком месте оказалась левая сторона - сдвигаем её левее или правее (теперь самый правый бит имеет номер least, а самый левый стоит на месте least+count2 \*/

if((count2-least)>0){

mask\_left = mask\_left >> (count2 - least);

}

else{

mask\_left = mask\_left << (least - count2 - (count%2));

}

/\* Правую сторону надо менять с левой местами - сдвигаем её всегда левее \*/

mask\_right = mask\_right << (count2 + least + (count%2));

/\* Вместо вырезанных бит вставляем обработанные (поменяли левую и правую сторону + отзеркалили их) \*/

original = (original & mask\_null) | mask\_left | mask\_right;

cout<<"New:"<<endl;

bin\_unsigned(original);

if(F\_or\_I == 0){

cout<<"New unsigned int: "<<original<<endl;

}

else{

fl\_new = \*((float\*)&original);

cout<<"New float: "<<fl\_new<<endl;

}

}

int main(){

unsigned int a = 0, b\_int = 0;

int count\_bit, least\_bit, work=10, flag\_error = 0;

float b = 0;

while(work!=0){

clear;

if((a!=0)||(b!=0)){

cout << "Unsigned int = " << a << endl <<"Float = " << b << endl;

cout << "Unsigned int = ";

bin\_unsigned(a);

cout << "Float = ";

bin\_unsigned(b\_int);

}

cout<<"1. Enter unsigned int and float"<<endl;

cout<<"2. Transform only unsigned int"<<endl;

cout<<"3. Transform only float"<<endl;

cout<<"4. Transform unsigned int and float"<<endl;

cout<<"0. Exit"<<endl;

cout<<"\nAnswer: ";

cin>>work;

flag\_error = 0;

if(work == 1){

cout<<"Enter unsigned int(>0 and !=0): ";

cin>>a;

cout<<"Enter float(!=0): ";

cin>>b;

if((a <= 0) || (b == 0)){

flag\_error = 1;

a = 0;

b = 0;

cout<<"Not correct numbers"<<endl;

}

}

if((work == 2) || (work == 3) || (work == 4)){

if((work == 2) && (a == 0)){

flag\_error = 1;

cout<<"You didn't set unsigned int"<<endl;

}

if((work == 3) && (b\_int == 0)){

flag\_error = 1;

cout<<"You didn't set float"<<endl;

}

if((work == 4) && ((a == 0) || (b\_int == 0))){

flag\_error = 1;

cout<<"You didn't set unsigned int or float"<<endl;

}

if(flag\_error == 0){

cout<<"Enter count of bits (>1): ";

cin>>count\_bit;

cout<<"Enter least bit: ";

cin>>least\_bit;

}

if(((least\_bit < 0) || (least\_bit >= 31)) && flag\_error == 0){

flag\_error = 1;

cout<<"Not correct least bit"<<endl;

}

if(((count\_bit <= 1) || ((least\_bit + count\_bit) > 32))&& flag\_error == 0){

flag\_error = 1;

cout<<"Not correct count of bits"<<endl;

}

}

if(flag\_error == 1){

work = 10;

}

switch(work){

case 1:

b\_int = \*((unsigned int\*)&b);

work = 10;

break;

case 2:

cout<<"\nAnswer unsigned int:"<<endl;

change(a, count\_bit, least\_bit, 0);

work = 10;

system("pause");

break;

case 3:

cout<<"\nAnswer float:"<<endl;

change(b\_int, count\_bit, least\_bit, 1);

work = 10;

system("pause");

break;

case 4:

cout<<"\nAnswer unsigned int:"<<endl;

change(a, count\_bit, least\_bit, 0);

cout<<"\nAnswer float:"<<endl;

change(b\_int, count\_bit, least\_bit, 1);

work = 10;

system("pause");

break;

case 0:

clear;

cout<<"Thanks for using"<<endl<<endl;

work = 0;

break;

default:

cout<<"Not correct"<<endl;

work = 10;

system("pause");

}

}

return 0;

}