void Push(Stack &S, char x)

{

PNode NewNode;

NewNode = new Node;

NewNode->a = x;

NewNode->next = S.Head;

NewNode->prev = NULL;

if (S.Head)

S.Head->prev = NewNode;

S.Head = NewNode;

if (!S.Tail) S.Tail = S.Head;

}

char Pop(Stack &S)

{

PNode TopNode = S.Head;

char x;

if (!TopNode)

return char(255);

x = TopNode->a;

S.Head = TopNode->next;

if (S.Head)

S.Head->prev = NULL;

else

S.Tail = NULL;

delete TopNode;

return x;

}

//сортировка по возрастанию чисел

int comp(const void\* a, const void\* b)

{

const firma\* k = (firma\*)a;

const firma\* m = (firma\*)b;

int s = ((k->sum) - (m->sum));

return s;

}

qsort(obj, 7, sizeof(firma), comp);

//сортировка по алфавиту

int comp(const void\* a, const void\* b)

{

int cmp(const void\* a, const void\* b)

{

return (strcmp((firma\*)a)->NameofaToy, ((firma\*)b)->NameofaToy);

}

}

qsort(obj, 7, sizeof(firma), comp);

void Sort(strahovka \*obj)

{

for (int k = 0; k < 5; k++)

{

for (int i = 0; i < 5 - k; i++)

{

if (strcmp(obj[i].fio, obj[i + 1].fio) > 0)

swap(obj[i], obj[i + 1]);

}

}

}

Лаб раб 7

#include <string.h>

#include <iostream>

using namespace std;

struct MyStruct

{ int H=0;

char A[4];

int B[4];

int C[4];

};

void GetData(MyStruct \*obj, int n)

{ cin.ignore();

for (int i = 0; i < n; i++)

{

cout << "шифр";

cin.getline(obj[i].A, 4);

cout<<endl;

cout << "план";

for (int r=0; r<4; r++)

{

cin >> obj[i].B[r];

cin.ignore();

}

cout<<"\t";

cout << "выполнено";

for (int r=0; r<4; r++)

{

cin >> obj[i].C[r];

cin.ignore();

}

}

}

void ShowData(MyStruct \*obj, int n)

{

for (int i = 0; i < n; i++)

{

cout<<"шифр:"<<" ";

cout << obj[i].A<<"\t";

cout<<"план:"<<" ";

for (int r=0; r<4; r++)

cout << obj[i].B[r]<<"\t";

cout<<"выполнено:"<<" ";

for (int r=0; r<4; r++)

cout << obj[i].C[r]<<"\t";

cout<< obj[i].H;

cout<<endl;

}

for(int k=0; k<n; k++)

{

for (int i=0; i<n-k; i++)

{

if (obj[i].H>obj[i+1].H)

swap(obj[i], obj[i+1]);

}

}

}

void Sortir(MyStruct \*obj, int n)

{

for (int i=0; i<n; i++)

{

for(int r=0; r<4; r++)

{

if (obj[i].C[r]>obj[i].B[r])

obj[i].H++;

}

}

}

int main()

{ int n;

cin>>n;

MyStruct \*obj = new MyStruct[n];

GetData(obj, n);

Sortir(obj,n);

ShowData(obj, n);

return 0;

}

лаб раб 9

ifstream in("vvod.txt");

if (in.is\_open());

else {

cout << "Error reading file" << endl;

return 1;

}

in >> SUM;

in >> percent;

while (!in.eof()) {

in >> sp[count].Surname >> sp[count].Fname >> sp[count].Lname >> sp[count].spentS >> sp[count].givenS;

count++;

}

in.close();

int rek(int\* Mass, int size) {

if (size > 0)

return Mass[size - 1] + Sum(Mass, size - 1);

else

return 0;

}

int Assem(int\* S, int n) {

int sum = 0;

\_asm {

mov ebx, S

xor esi,esi

mov eax, sum

mov ecx, n

loop1: add eax, [ebx][esi]

add esi,4

loop loop1

mov sum,eax

}

return sum;

}

ofstream outf("vyvod.txt");

if (outf.is\_open());

else {

cout << "Error opening the file" << endl;

return 2;

}

for (int i = 0; i < count; i++) {

outf << sp[i].Surname << " " << sp[i].Fname << " " << sp[i].Lname << " " << sp[i].spentS << " " << sp[i].NgivSum << " " << sp[i].sumD << endl;

}

outf.close();

qsort(sp, count, sizeof(spending), cmp\_Sur);

for (int i = 0; i < count; i++) {

cout << sp[i].Surname << sp[i].Fname << sp[i].Lname << sp[i].spentS << sp[i].givenS << endl;;

}

cin >> str;

int\* man = (int\*)bsearch(&str, sp, count, sizeof(spending), cmpS);

if (man != NULL) {

for (int i = 0; i < count; i++) {

if (!strcmp(str, sp[i].Surname))

cout << sp[i].Surname << " " << sp[i].Fname << " " << sp[i].Lname << " " << sp[i].spentS << " " << sp[i].NgivSum << " " << sp[i].sumD << endl;

}

}