ФАКУЛТЕТ ЗА ЕЛЕКТРОТЕХНИКА И ИНФОРМАЦИСКИ ТЕХНОЛОГИИ

ПОДАТОЧНИ СТУКТУРИ И ПРОГРАМИРАЊЕ 2023/2024

АУДИТОРИСКИ ВЕЖБИ 4

МАГАЦИНИ (ВО C И C++)

ЗАДАЧИ:

1. Да се напише функција која како аргумент добива декаден број и променлива од тип магацин. Функцијата треба да го испечати бинарниот број, но со користење само на магацинот. Да се напишат сите структури и функции кои се потребни за да се реализира работата на оваа функција. Во главната програма да се провери работата на функцијата.

Решение:

```
#include <iostream>
using namespace std;

int const MAX = 20;

struct element
{
	int info;
};

struct magacin
{
	element niza[MAX];
	int vrv;
	void inicijaliziraj();
	bool ePrazen();
	bool ePoln();
	void vmetni(int e);
	element izvadi();
};

void magacin::inicijaliziraj()
{
	vrv = -1;
}

bool magacin::ePrazen()
{
	return(vrv == -1);
}

bool magacin::ePoln()
{
	return (vrv == MAX - 1);
}

void magacin::vmetni(int e)
{
	if (ePoln())
	{
		cout << "Magacinot e poln i ne mozhe da se vmetnuvaat elementi!" << endl;
		return;
	}

	niza[++vrv].info = e;
}

element magacin::izvadi()
{
	if (ePrazen())
	{
		cout << "Magacinot e prazen i ne mozhe da se vadat elementi!" << endl;
		return{};
	}
	return niza[vrv--];
```

```
}

void DekVoBin(int dek, magacin &m)
{
	int cif = dek;
	m.inicijaliziraj(); //pred da vmetnuvame elementi vo magacinot, mora da go
inicijalizirame!!!
	while (dek != 0) //go zemame ostatokot i go smestuvame vo magacin
	{
		m.vmetni(dek % 2);
		dek /= 2;

	}
	//kako gi vadime elementite od magacinot, taka i gi pechatime
	//so shto go dobivame binarniot ekvivalent na dek
	cout << "Binarniot ekvivalent na " << cif << " e ";
	while (!(m.ePrazen()))
	{
		cif = m.izvadi().info;
		cout << cif;
	}
	cout << endl;
}

int main()
{
	magacin mag;
	int dek;
	cout << "Vnesete dekaden broj: ";
	cin >> dek;
	DekVoBin(dek, mag);
}
```

2. Да се провери дали во даден аритметички израз заградите се балансирани.

Пример: Балансирани загради {1+[2–5]+(2+{6*3})/1}
Небалансирани загради {1+[2–5}+(2+{6*3})/1}

Решение:

```
#include <iostream>
#include <string>
using namespace std;
int const MAX = 20;

struct element
{
	char info; /* за магацинот да работи со знаци, единствено го менуваме овој дел!!!
*/
};

struct Stack
{
	element array[MAX];
	int top;
	void init();
	bool isEmpty();
	bool isFull();
	void push(element e);
	element pop();
	element peek();
};
```

```
void Stack::init()
{
	top = -1;
}

bool Stack::isEmpty()
{
	return(top == -1);
}

bool Stack::isFull()
{
	return (top == MAX - 1);
}


void Stack::push(element e)
{
	if (isFull())
	{
		cout << "Magacinot e poln i ne mozhe da se vmetnuvaat elementi!" << endl;
		return;
	}

	array[++top] = e;
}

element Stack::pop()
{
	if (isEmpty())
	{
		cout << "Magacinot e prazen i ne mozhe da se vadat elementi!" << endl;
		return{};
	}
	return array[top--];
}

element Stack::peek()
{
	if (isEmpty())
	{
		cout << "Magacinot e prazen i ne mozhe da se vadat elementi!" << endl;
		return{};
	}
	return array[top];
}

bool proveriZagradi(string izraz, Stack &magacin)
{
	magacin.init();
	for (int i = 0; i < izraz.length(); i++) {
		if (izraz[i] == '{' || izraz[i] == '[' || izraz[i] == '(') {
			element e;
			e.info = izraz[i];
			magacin.push(e);
		}
		else if (izraz[i] == '}')
			if (magacin.peek().info == '{')
				magacin.pop();
			else return false;
		else if (izraz[i] == ']')
			if (magacin.peek().info == '[')
				magacin.pop();
			else
				return false;
```

```
            else if (izraz[i] == ')')
                  if (magacin.peek().info == '(')
                        magacin.pop();
                  else
                        return false;
      }
      if (magacin.isEmpty())
            return true;
      else
            return false;
}

int main()
{
      Stack magacin;
      string izraz;
      cout << "Vnesete aritmetichki izraz:" << endl;
      cin >> izraz;
      if (proveriZagradi(izraz, magacin))
            cout << "Izrazot " << izraz << " ima balansirani zagradi";
      else
            cout << "Izrazot " << izraz << " nema balansirani zagradi";
}
```

3. Да се напише функција која како аргументи добива магацин со веќе внесени елементи и еден цел број. Во магацинот да се најде последното појавување на елементот кој има вредност еднаква со вториот аргумент. Ако постои елемент со таква вредност, истиот да се отстрани од магацинот и да се врати позицијата на која е најден (броејќи од горе надолу). Доколку не се најде елемент со таква вредност, функцијата да врати вредност нула. Да се напише главна програма во која ќе се внесат вредностите на еден магацин и број за пребарување и ќе се повика функцијата.
   Забелешка: Како дополнителна структура може да се користи само еден магацин. Да се користат операциите за додавање и отстранување на елементи од магацин.

Пример:

Пред промена: **Магацинот е:** **|10|** **|2|** **|10|** **|7|** **|10|** **|10|** **|9| број = 10**	**После промената:** **Магацинот ќе биде:** **|10|** **|2|** **|10|** **|7|** **|10|** **|9|** **Позиција е 6**

Решение:

```
#include <iostream>
#include <string>
using namespace std;

#define MAX 10

struct stack {
    int array[MAX];
    int top;
    void init();
    bool isEmpty();
```

```
    bool isFull();
    void push(int e);
    int pop();
    int peek();
    void print();
};

void stack::print(){
    for(int i =0;i<=top;i++)
        cout<<array[i]<<" ";
}

void stack::init() {
    top = -1;
}

bool stack::isEmpty() {
    return (top == -1);
}

bool stack::isFull() {
    return (top == MAX - 1);
}

void stack::push(int e) {
    if (isFull()) {
        cout << "magacinot e poln." << endl;
        return;
    }
    array[++top] = e;
}

int stack::pop() {
    if (isEmpty()) {
        cout << "magacinot e prazen." << endl;
        exit(-1);
    }
    return array[top--];
}

int stack::peek() {
    if (isEmpty()) {
        cout << "magacinot e prazen." << endl;
        exit(-1);
    }
    return array[top];
}
int funkcija(stack &s, int b){
    stack d;int poz=0;
    d.init();
    while(!s.isEmpty()){
        d.push(s.pop());
    }
```

```
    while(!d.isEmpty()){
     int elem = d.peek(); // ja zemame vrednosta bez da go otstranime
     if(elem==b && poz==0){
        poz = d.top+1;// dodavame +1 zatoa sho top pocnuva od 0, a pozicijata od 1
     }
     else {
        s.push(elem); // ja dodavame
     }
     d.pop();
    }
    return poz;
}
int main() {
    stack m1;
    int broj = 10;
    m1.init();

    m1.push(9);
    m1.push(10);
    m1.push(10);
    m1.push(7);
    m1.push(10);
    m1.push(2);
    m1.push(10);// gi dodavame vo obraten redosled od primerot za da se kreiraat vo
tocniot redosled
    int p = funkcija(m1,broj);
    if(p==0){
    cout<<"Elementot ne e pronajden"<<endl;
    }else {
        cout << "Elementot e  na pozicija "<< p<< endl;
    }

    while (!m1.isEmpty()) {
        cout << m1.pop() << endl;
    }

    return 0;
}
```

4. Да се напише функција која како аргумент добива магацин во кој се наоѓаат букви (овој магацин треба да се модифицира во функцијата). Во функцијата треба да се споредат буквите со буквите кои се наоѓаат на иста позиција во превртен магацин и доколку две букви во ASCII се разликуваат за 1 во излезниот магацин да се запише помалата буква. Во функцијата може да се користат помошни магацини (потребни се минимум два помошни магацини), но користење на низи не е дозволено. Дополнително во програмата да се дефинира структурата магацин заедно со сите функции кои се потребни за функционирање на магацинот: за иницијализација, за проверка дали магацинот е полн и дали е празен, за внесување елементи, за изнесување на елементи, за гледање на елементот кој е најгоре во магацинот и за печатење на елементите без вадење на истите од магацинот. Излезниот магацин од функцијата се печати во главната функција.

Пример:

Пред промена: **Магацинот е:** **|p|** **|o|** **|s|** **|t|** **|a|** **|q|** **Прво внесен елемент во магацинот е p, а последно внесен е q.**	**После промената:** **Магацинот ќе биде:** **|p|** **|s|** **|s|** **|p|** **Превртениот магацин ќе ги има елементите |q||a||t||s||o||p|, па разлика од 1 има помеѓу паровите елементи |p| и |q| (од каде се добива |p|) и |s| и |t| (од каде се довива |s|)**

Решение:

```
#include <iostream>
#include <string>
#include <stdlib.h>
using namespace std;

#define MAX 10

struct stack {
	char array[MAX];
	int top;
	void init();
	bool isEmpty();
	bool isFull();
	void push(char e);
	char pop();
	char peek();
	void print();
};

void stack::print(){
	for(int i =0;i<=top;i++)
		cout<<array[i]<<" ";

	cout<<endl;
}

void stack::init() {
	top = -1;
}

bool stack::isEmpty() {
	return (top == -1);
}

bool stack::isFull() {
	return (top == MAX - 1);
}

void stack::push(char e) {
	if (isFull()) {
		cout << "magacinot e poln." << endl;
		return;
	}
	array[++top] = e;
}

char stack::pop() {
	if (isEmpty()) {
		cout << "magacinot e prazen." << endl;
```

```
                exit(-1);
        }
        return array[top--];
}

char stack::peek() {
        if (isEmpty()) {
                cout << "magacinot e prazen." << endl;
                exit(-1);
        }
        return array[top];
}

void pronajdi(stack *m1) {
        stack help1, help2;
        help1.init();
        help2.init();

        while(!m1->isEmpty()){
                char c = m1->pop();
                help1.push(c);
                help2.push(c);
        }

        while(!help2.isEmpty())
                m1->push(help2.pop());

        while(!m1->isEmpty()){
                char c1 = m1->pop();
                char c2 = help1.pop();

                if(c1-c2 == 1){
                        help2.push(c2);
                }

                if(c2-c1 == 1){
                        help2.push(c1);
                }
        }

        while(!help2.isEmpty())
                m1->push(help2.pop());


}

int main() {
        stack m1;

        m1.init();

        m1.push('q');
        m1.push('a');
        m1.push('t');
        m1.push('s');
        m1.push('o');
        m1.push('p');

        pronajdi(&m1);

        while (!m1.isEmpty()) {
                cout << m1.pop() << endl;
        }

        return 0;
}
```

5. Да се реализира лото игра со помош на магацин. За зачувување на учесниците се користи магацин, така што секој од елементите на магацинот го содржи името и презимето на учесникот во играта, како и низа од 7 броеви кои корисникот ги внел во своето добитно ливче. Во главната програма се дадени и магацинот учесници и добитното ливче. Да се напише функција која ќе ги измине сите учесници во лото играта, од нив во посебен магацин ќе ги издвои оние кои се добитници на лото 7 (сите 7 броеви на корисникот се совпаѓаат со добитната комбинација) и добиениот магацин од добитници, кој ќе биде проследен во главната функција за печатење.

Пример:

Учесници:

Петко Петковски	9, 23, 13, 25, 32, 21, 11
Милан Милановски	32, 25, 13, 21, 12, 10, 5
Ангела Ангелковска	9, 25, 13, 11, 32, 21, 23
Стојан Стојановски	23, 11, 17, 9 , 32, 25, 21

Добитна комбинација: 9, 23, 11, 32, 25, 13, 21

Добитници:

Петко Петковски	9, 23, 13, 25, 32, 21, 11
Ангела Ангелковска	9, 25, 13, 11, 32, 21, 23

Решение:

```
#include <iostream>
#include <string>
#include <stdlib.h>

using namespace std;

#define MAX 10

struct element {
	string imeprezime;
	int livche[7];
};

struct stack {
	element array[MAX];
	int top;
	void init();
	bool isEmpty();
	bool isFull();
	void push(element e);
	element pop();
	element peek();
};

void stack::init() {
	top = -1;
}

bool stack::isEmpty() {
	return (top == -1);
}

bool stack::isFull() {
	return (top == MAX - 1);
}

void stack::push(element e) {
	if (isFull()) {
		cout << "magacinot e poln." << endl;
		return;
```

```
        }
        array[++top] = e;
}

element stack::pop() {
        if (isEmpty()) {
                cout << "magacinot e prazen." << endl;
                exit(-1);
        }
        return array[top--];
}

element stack::peek() {
        if (isEmpty()) {
                cout << "magacinot e prazen." << endl;
                exit(-1);
        }
        return array[top];
}

void sedumka(int *dobitna, stack ucesnici, stack *dobitnici) {
        element ucesnik;
        int countcheck;
        bool elementcheck;

        while (!ucesnici.isEmpty()) {
                countcheck = 0;
                ucesnik = ucesnici.pop();

                for (int i = 0; i < 7; i++) {
                        elementcheck = false;
                        for (int j = 0; j < 7; j++) {
                                if (ucesnik.livche[i] == dobitna[j]) {
                                        elementcheck = true;
                                }
                        }
                        if (elementcheck) {
                                countcheck += 1;
                        }
                }

                if (countcheck == 7) {
                        dobitnici->push(ucesnik);
                }
        }
}

int main() {
        stack ucesnici, dobitnici;
        element e;
        int dobitna[7] = { 9, 23, 11, 32, 25, 13, 21 };

        ucesnici.init();
        dobitnici.init();

        e.imeprezime = "Petko Petkovski";
        e.livche = { 9, 23, 13, 25, 32, 21, 11 };
        ucesnici.push(e);
        e.imeprezime = "Milan Milanovski";
        e.livche = { 32, 25, 13, 21, 12, 10, 5 };
        ucesnici.push(e);
        e.imeprezime = "Angela Angelkovska";
        e.livche = { 9, 25, 13, 11, 32, 21, 23 };
        ucesnici.push(e);
        e.imeprezime = "Stojan Stojanovski";
        e.livche = { 23, 11, 17, 9, 32, 25, 21 };
        ucesnici.push(e);
        sedumka(dobitna, ucesnici, &dobitnici);
```

```
	while (!dobitnici.isEmpty()) {
		cout << dobitnici.pop().imeprezime << endl;
	}

	return 0;
}
```

6. Да се напише програма во која ќе се работи со магацини. Еден магацин се состои од повеќе елементи. Во елементите основен податочен тип е цел број. Треба да се направи промена на насоката на броевите во магацинот помеѓу два елементи со вредност нула. Да се претпостави дека во влезниот магацин секогаш има две нули и помеѓу нив секогаш има елементи. Во програмата е дозволена употреба на дополнителни магацини. Да се дефинира структурата на еден магацин со основен елемент во кој има цел број. Да се дефинираат потребните функции за правилна работа на магацинот- иницијализација, додавање нова вредност во магацинот, повлекување постоечка вредност од магацинот, читање на првата вредност во магацинот, проверка дали магацинот е полн и проверка дали магацинот е празен. Употребата на низи или редови не е дозволена.

Пример:

Во магацинот се чуваат следните податоци:	По промената магацинот изгледа вака:
1 3 5 **0** **3** **4** **5** **6** **0** 1 3	1 3 5 **0** **6** **5** **4** **3** **0** 1 3

Решение:

```
#include <iostream>
using namespace std;
#define MAX 20

struct element
{
	int info;
};

struct magacin
{
	element niza[MAX];
	int vrv;
	void inicijaliziraj();
	bool ePrazen();
	bool ePoln();
	void vmetni(int e);
	element izvadi();
};
```

```
void magacin::inicijaliziraj()
{
	vrv = -1;
}

bool magacin::ePrazen()
{
	return(vrv == -1);
}

bool magacin::ePoln()
{
	return (vrv == MAX - 1);
}

void magacin::vmetni(int e)
{
	if (ePoln())
	{
		cout << "Magacinot e poln i ne mozhe da se vmetnuvaat elementi!" << endl;
		return;
	}

	niza[++vrv].info = e;
}

element magacin::izvadi()
{
	if (ePrazen())
	{
		cout << "Magacinot e prazen i ne mozhe da se vadat elementi!" << endl;
		return{};
	}
	return niza[vrv--];
}

void proveri(magacin& mag) {
	magacin pomosen_magacin_1, pomosen_magacin_2, pomosen_magacin_3;
	pomosen_magacin_1.inicijaliziraj();
	pomosen_magacin_2.inicijaliziraj();
	pomosen_magacin_3.inicijaliziraj();
	int izvadena_vrednost = 1;
	while (izvadena_vrednost!=0) {
		izvadena_vrednost = mag.izvadi().info;
		pomosen_magacin_1.vmetni(izvadena_vrednost);
		if (izvadena_vrednost == 0) {
			pomosen_magacin_1.izvadi();
		}
	}
	izvadena_vrednost = 1;
	while (izvadena_vrednost != 0) {
		izvadena_vrednost = mag.izvadi().info;
		pomosen_magacin_2.vmetni(izvadena_vrednost);
		if (izvadena_vrednost == 0) {
			pomosen_magacin_2.izvadi();
		}
	}
	while (!pomosen_magacin_2.ePrazen()) {
		izvadena_vrednost = pomosen_magacin_2.izvadi().info;
		pomosen_magacin_3.vmetni(izvadena_vrednost);
	}
	mag.vmetni(0);
	while (!pomosen_magacin_3.ePrazen()) {
		izvadena_vrednost = pomosen_magacin_3.izvadi().info;
		mag.vmetni(izvadena_vrednost);
	}
	mag.vmetni(0);
	while (!pomosen_magacin_1.ePrazen()) {
```

```
			izvadena_vrednost = pomosen_magacin_1.izvadi().info;
			mag.vmetni(izvadena_vrednost);
		}
		while (!mag.ePrazen()) {
			cout<<mag.izvadi().info<<endl;
		}
}

int main()
{
		magacin mag;
		mag.inicijaliziraj();
		int a[11] = { 1,3,5,0,3,4,5,6,0,1,3 };

		for (int i = 0; i <11; i++) {
			mag.vmetni(a[i]);
		}
		while (!mag.ePrazen()) {
			cout << mag.izvadi().info << endl;
		}
		for (int i = 0; i < 11; i++) {
			mag.vmetni(a[i]);
		}
		cout << endl;
		proveri(mag);
		return 0;
}
```

7. Да се напише функција која како аргумент добива магацин (овој магацин треба да се модифицира во функцијата). Во функцијата треба да се проверат сите броеви кои се елементи од магацинот и секој број да се спореди со неговите соседни броеви. Притоа треба да се утврди дали соседните броеви може да бидат делители на разгледуваниот број. Ако може, тогаш во излезниот магацин се запишува количникот (ако и двата соседни броеви може да бидат делители, тогаш се запишуваат двата количници). Во функцијата може да се користат помошни магацини (потребен е минимум еден помошен магацин), но користење на низи не е дозволено. Дополнително во програмата да се дефинира структурата магацин заедно со сите функции кои се потребни за функционирање на магацинот: за иницијализација, за проверка дали магацинот е полн и дали е празен, за внесување елементи, за изнесување на елементи, за гледање на елементот кој е најгоре во магацинот и за печатење на елементите без вадење на истите од магацинот. Излезниот магацин од функцијата се испечати во главната функција.

Пример:

Пред промена: **Магацинот е:** **|1|** **|2|** **|5|** **|10|** **|2|** **Прво внесен елемент во магацинот е 2, а последно внесен е 1.**	**После промената:** **Магацинот ќе биде:** **|2|** **|2|** **|5|** **На елементот 2 соседот 1 може да му биде делител, па се добива 2 во излезот, на бројот 10 делител може да му биде 5 (така се добива вториот елемент 2), а може да му биде делител и 2 (така се добива елементот 5)**

Решение:

```
#include <iostream>
#include <string>
using namespace std;
```

```
#define MAX 10

struct stack {
	int array[MAX];
	int top;
	void init();
	bool isEmpty();
	bool isFull();
	void push(int e);
	int pop();
	int peek();
	void print();
};

void stack::print(){
	for(int i =0;i<=top;i++)
		cout<<array[i]<<" ";
}

void stack::init() {
	top = -1;
}

bool stack::isEmpty() {
	return (top == -1);
}

bool stack::isFull() {
	return (top == MAX - 1);
}

void stack::push(int e) {
	if (isFull()) {
		cout << "magacinot e poln." << endl;
		return;
	}
	array[++top] = e;
}

int stack::pop() {
	if (isEmpty()) {
		cout << "magacinot e prazen." << endl;
		exit(-1);
	}
	return array[top--];
}

int stack::peek() {
	if (isEmpty()) {
		cout << "magacinot e prazen." << endl;
		exit(-1);
	}
	return array[top];
}

void promeni(stack &m1) {
	int count = 0, prev = 0;
	int mom, next;
	stack m2;
	m2.init();
	while (!m1.isEmpty()) {
		mom = m1.pop();
		if(count!=0)
			if(mom%prev==0)
				m2.push(mom/prev);

		if(!m1.isEmpty()){
```

```
                        next = m1.peek();
                        if(mom%next==0)
                                m2.push(mom/next);
                }
                prev = mom;
                count++;
        }

        while (!m2.isEmpty()) {
                m1.push(m2.pop());
        }
}

int main() {
        stack m1;

        m1.init();

        m1.push(2);
        m1.push(10);
        m1.push(5);
        m1.push(2);
        m1.push(1);

        promeni(m1);

        while (!m1.isEmpty()) {
                cout << m1.pop() << endl;
        }

        return 0;
}
```