Учреждение образования

„Белорусский государственный университет информатики и радиоэлектроники”

Кафедра «Вычислительных методов и программирования»

ОТЧЕТ

По лабораторным работам №6-7

«Стеки. Двухсвязные списки»

Выполнила:

Студентка АСОИ

Группы №820605

ФИО

Вариант № 9

Проверил:

ассистент кафедры ВМИП

Беспалов С.А.

Минск 2019

1. Индивидуальное задание:

В. *Создать стек, состоящий из n целых чисел. Выполнить задание. Информационную часть в оперативной памяти не перемещать. Результат вывести на экран. В конце работы освободить всю динамически выделенную память.*

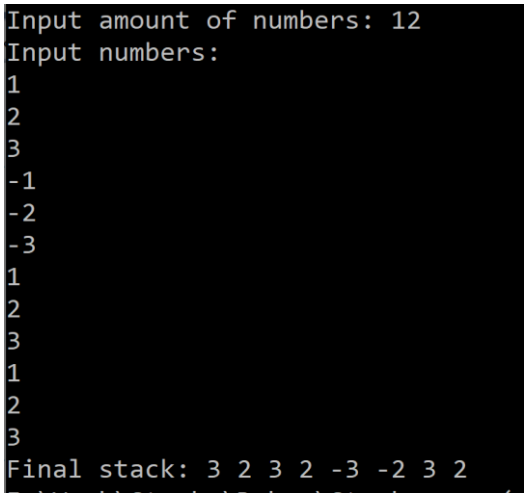
В9. Удалить каждый третий элемент стека.

Текст программы:

```
#include "pch.h"
#include <iostream>
using namespace std;

struct stacks
{
	int inf;
	stacks* address;
} numbers;
stacks *addstacks(stacks*, int);
stacks* delthirdstack(stacks*);
stacks* readstack(stacks*);

int main()
{
	stacks* first = NULL;
	int num, amount;
	cout << "Input amount of numbers: "; cin >> amount;
	cout << "Input numbers: " << endl;
	for (int i = 0; i < amount; i++)
	{
		cin >> num;
		first = addstacks(first, num);
	};
	stacks* workaddress = first->address;
	while (workaddress != NULL && workaddress->address != NULL )
	{
		if((workaddress = delthirdstack(workaddress))!= NULL)
			workaddress=workaddress->address;
	}
	cout << "Final stack: ";
	while (first!=NULL)
	first = readstack(first);
	return 0;
}

stacks *addstacks(stacks* first, int num)
{
	stacks *newaddress = new stacks;
	newaddress->inf = num;
	newaddress->address = first;
	return newaddress;
}

stacks* delthirdstack(stacks* address)
{
	stacks* third = address->address;
	address->address = address->address->address;
	delete third;
	return (address->address);
}
```

```
stacks* readstack(stacks*a)
{
	stacks* del = a;
	cout << a->inf << " ";
	a = a->address;
	delete del;
	return a;
}
```

Результат работы программы:



2. Индивидуальное задание:

В. *Выполнить задание в соответствии с вариантом. Информационную часть в оперативной памяти не перемещать. Результат вывести на экран. В конце работы освободить всю динамически выделенную память.*

В9. Создать двусвязанный список, состоящий из n целых чисел. Отрицательные элементы удалить, а четные перенести во второй список.

Текст программы:

```
#include "pch.h"
#include <iostream>
#include <math.h>

using namespace std;

struct bidirlist
{
	int inf;
	bidirlist *left;
	bidirlist *right;
} *sl, *sr, *sl2, *sr2, *workaddress, *workaddress2;

void NewQueue(bidirlist**, bidirlist**);
bidirlist* AddQueueRight(bidirlist*,int);
bidirlist* DelQueue(bidirlist*);
void ReadAndDelQueueAll(bidirlist**, bidirlist**);

int n, inf;

int main()
{
```

```
	NewQueue(&sl, &sr); workaddress = sl;
	cout << "Input amount of numbers: "; cin >> n;
	cout << "Input numbers" << endl;
	for (int i = 0; i < n; i++)
	{
		cin >> inf;
		workaddress = AddQueueRight(workaddress,inf);
	}
	NewQueue(&sl2, &sr2); workaddress2 = sl2;
	workaddress = sl->right;
	while (workaddress!= sr)
	{
		if ((workaddress->inf) < 0)
			workaddress = DelQueue(workaddress);
		else if (fmod((workaddress->inf), 2) == 0)
		{
			workaddress2 = AddQueueRight(workaddress2, workaddress->inf);
			workaddress = DelQueue(workaddress);
		}
		else workaddress = workaddress->right;
	}
	cout << "The first queue (only odd positive numbers) : ";
	if (sl->right != sr) ReadAndDelQueueAll(&sl, &sr);
	else cout << "not existing.";
	cout << endl << "The second queue (only even positive numbers) : ";
	if (sl2->right != sr2) ReadAndDelQueueAll(&sl2, &sr2);
	else cout << "not existing.";
	cout << endl;
	return 0;
}

void NewQueue(bidirlist** sl, bidirlist** sr)
{
	*sl = new bidirlist;
	*sr = new bidirlist;
	(*sl)->left = NULL;
	(*sl)->right = *sr;
	(*sr)->left = *sl;
	(*sr)->right = NULL;
	return;
}

bidirlist* AddQueueRight(bidirlist* first, int inf)
{
	bidirlist *add = new bidirlist;
	add->inf = inf;
	add->left = first;
	add->right = first->right;
	add->right->left = add;
	first->right = add;
	return add;
}

bidirlist* DelQueue(bidirlist* del)
{
	bidirlist* next = del->right;
	del->left->right = del->right;
	del->right->left = del->left;
	delete del;
	del = NULL;
	return next;
}

void ReadAndDelQueueAll(bidirlist **l, bidirlist **r)
{
```

```
	workaddress = (*l)->right;
	while (workaddress != *r)
	{
		cout << workaddress->inf << " "; workaddress = DelQueue(workaddress);
	}
	delete *l; *l = NULL;
	delete *r; *r = NULL;
	return;
}
```

Результат работы программы:

```
Input amount of numbers: 10
Input numbers
1
-2
3
5
8
10
-45
0
23
1
The first queue (only odd positive numbers) : 1 3 5 23 1
The second queue (only even positive numbers) : 8 10 0
```