Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра «Вычислительных методов и программирования»

ОТЧЕТ

По лабораторной работе №11

«Стеки. Двусвязные списки»

Выполнил:

Студент АСОИ гр. №820605

ФИО

Вариант № 6

Проверил:

Ассистент кафедры ВМИП

Беспалов С.А.

Минск 2019

1. **Задание:**

B. *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.*

В6. Добавить элемент со значением 77 перед предпоследним элементом стека.

Текст программы:

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

struct st {
	int inf;
	st *adr;
};

st *DeleteStack(st *);
st *AddToStack(int, st *);
void Add77(st *);

int main() {
	int i, n, x;
	st *head=NULL;
	cout << "Enter the number of elements" << endl;
	cin >> n;
	for (i = 0; i < n; i++) {
		cin >> x;
		head = AddToStack(x, head);
	}
	Add77(head);
	DeleteStack(head);
	system("pause");
	return 0;
}

st *AddToStack(int x, st *v) {
	st *top = new st;
	top->inf = x;
	top->adr = v;
	return top;
}

st *DeleteStack(st *v) {
	st *top;
	while (v != NULL)
	{
		top = v;
		cout << v->inf << endl;
		v = v->adr;
		delete top;
	}
	return NULL;
}

void Add77(st *head) {
	st *top=head;
	st *v = new st;
	if (top->adr == NULL) {
		v->inf=top->inf;
		v->adr = NULL;
		top->inf = 77;
```

```
                top->adr = v;
        }
        else {
                while (top->adr->adr != NULL)
                {
                        top = top->adr;
                }
                v->inf = 77;
                v->adr = top->adr;
                top->adr = v;
        }
}
```

2. **Задание:**

B. *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.*

В6. Создать два двусвязанных списка, состоящих из *n* целых чисел, упорядоченных по неубыванию. Переместить все данные в третий список, удаляя повторяющиеся значения.

Текст программы:

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

struct list {
        int info;
        list *pred, *sled;
};

void CreateFrame(list **, list **);
void AddElement(list *, int);
void DeleteList(list **, list **);
void DeleteElement(list *);

int main() {
        int i, j, n, m, k, info, x;
        list *first1, *last1, *first2, *last2, *first3, *last3;
        cout << "Creation the first list" << endl;
        CreateFrame(&first1, &last1);
        cin >> n;
        for (i = 0; i < n; i++) {
                cin >> info;
                AddElement(last1, info);
        }
        cout << "Creation the second list" << endl;
        CreateFrame(&first2, &last2);
        cin >> m;
        for (j = 0; j < m; j++) {
                cin >> info;
                AddElement(last2, info);
        }

        CreateFrame(&first3, &last3);
        while (first1->sled != last1 && first2->sled != last2) {
                if (first1->sled->info < first2->sled->info) {
                        x = first1->sled->info;
                        AddElement(last3, first1->sled->info);
                        DeleteElement(first1->sled);
                }
                else {
```

```
                        x = first2->sled->info;
                        AddElement(last3, first2->sled->info);
                        DeleteElement(first2->sled);
                }
                while (first1->sled->info == x) DeleteElement(first1->sled);
                while (first2->sled->info == x) DeleteElement(first2->sled);
        }
        while (first1->sled != last1)
                if (first1->sled->info != x) {
                        x = first1->sled->info;
                        AddElement(last3, first1->sled->info);
                        DeleteElement(first1->sled);
                }
                else DeleteElement(first1->sled);
        while (first2->sled != last2)
                if (first2->sled->info != x) {
                        x = first2->sled->info;
                        AddElement(last3, first2->sled->info);
                        DeleteElement(first2->sled);
                }
                else DeleteElement(first2->sled);
        cout << "------" << endl;
        DeleteList(&first3, &last3);
        system("pause");
        return 0;
}

void CreateFrame(list **first, list **last) {
        *first = new list;
        *last = new list;
        (*first)->pred = (*last)->sled = NULL;
        (*first)->sled = *last;
        (*last)->pred = *first;
}

void AddElement(list *top, int info) {
        list *current = new list;
        current->info = info;
        current->sled = top;
        current->pred = top->pred;
        top->pred->sled = current;
        top->pred = current;
}

void DeleteElement(list *top) {
        top->pred->sled = top->sled;
        top->sled->pred = top->pred;
        delete top;
}

void DeleteList(list **first, list **last) {
        list *top = (*first)->sled;
        while (top != *last) {
                cout << top->info << ' ' << endl;
                DeleteElement(top);
                top = (*first)->sled;
        }
        delete *first; *first = NULL;
        delete *last; *last = NULL;
}
```