Министерство образования Республики Беларусь

Учреждение образования «Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №5
по ООПиП
«ПЕРЕГРУЗКА ОПЕРАЦИЙ”

Выполнила: студентка 2-го курса

группы АС-53 Замулко Д.И.

Проверила: Давидюк Ю.И.

Брест, 2020

Лабораторная работа №5
«ПЕРЕГРУЗКА ОПЕРАЦИЙ”

Цель: получить практические навыки создания абстрактных типов данных и перегрузки операций в языке C++.

Вариант 11

АДТ – однонаправленный список с элементами типа **char.** Дополнительно перегрузить следующие операции:

[] – доступ к элементу в заданной позиции, например:

int i; char c;

list L;

c=L[i];

+ – объединить два списка;

!= – проверка на неравенство.

Код:

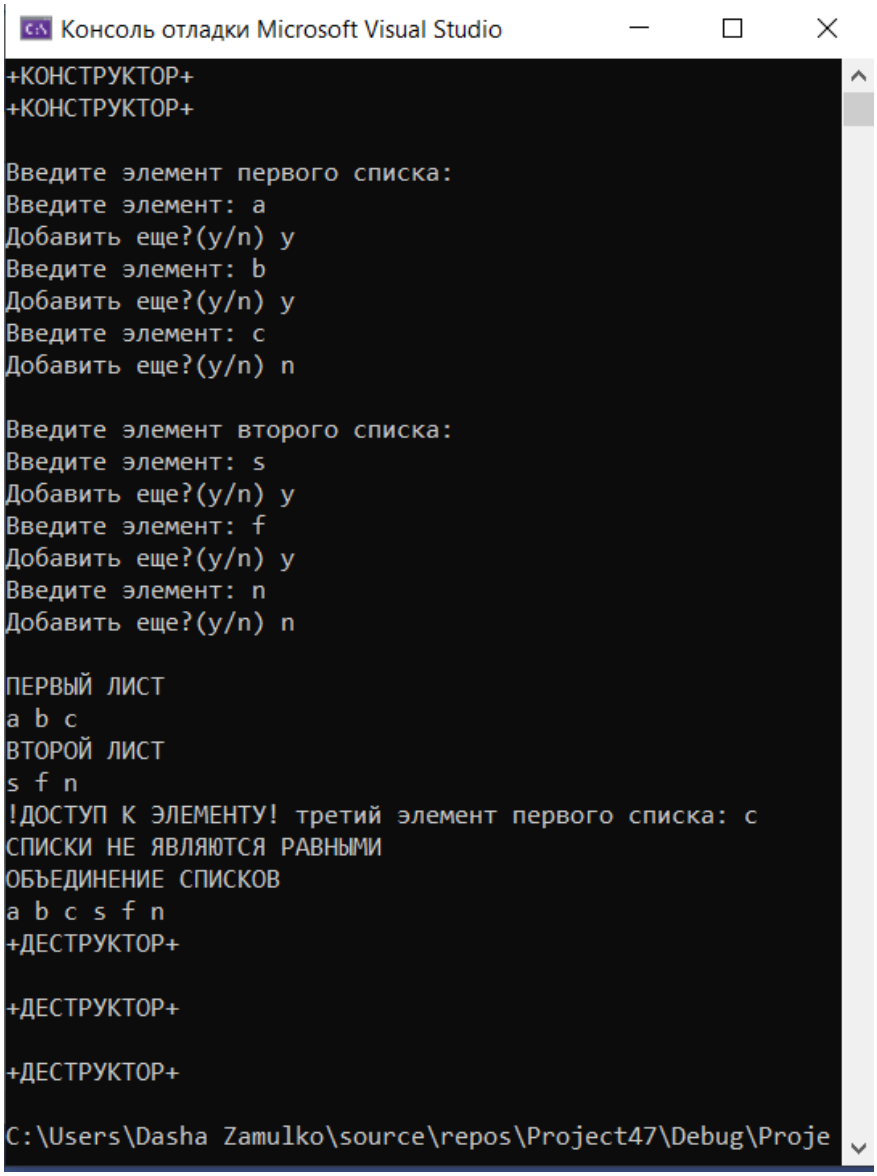
```
#include <iostream>
#include <cstring>

using namespace std;

struct Element {
	char data;
	Element* next;
};

const int MAX = 15;

class List {

	Element* pHead;
	Element* pPrev;
	int size = 0;

public:

	List();
	~List();
	void addToList(char data);
	void printList();
	Element* operator [](int i);
	friend List operator +(List&, List&);
	friend bool operator !=(List&, List&);
};

List::List() {
	pHead = NULL;
	pPrev = NULL;
	cout << "+КОНСТРУКТОР+" << endl;
}List::~List() {
	cout <<endl<< "+ДЕСТРУКТОР+" << endl;
}
```

```
void List::addToList(char data) {
	Element* temp = new Element;
	if (pHead == NULL) {
		pHead = temp;
		temp->data = data;
		temp->next = NULL;
		pPrev = temp;
		size++;
	}
	else if (size < MAX) {
		pPrev->next = temp;
		temp->data = data;
		temp->next = NULL;
		pPrev = temp;
		size++;
	}
	else {
		cout << "Максимальный размер достигнут" << endl;
	}
}

void List::printList() {
	Element* pTemp = pHead;
	while (pTemp != NULL) {
		std::cout << pTemp->data << " ";
		pTemp = pTemp->next;
	}
}

Element* List::operator [](int i) {
	if (i < 0 && i > MAX) {
		cout << "Неверный индекс" << endl;
	}
	else {
		Element* key1 = this->pHead;
		Element* key2;
		for (int j = 0; j < i; j++) {
			key2 = key1->next;
			key1 = key2;
		}
		return key1;
	}
}

List operator +(List& l1, List& l2) {

	Element* temp = new Element;
	int h = l1.size;

	for (int i = h; i < MAX; i++) {

		if (l2.pHead != NULL) {
			char a = l2.pHead->data;
			l1.addToList(a); Element* pTemp = l2.pHead;
			pTemp = pTemp->next;
			l2.pHead = pTemp;
			l2.size--;

		}

		else {

			return l1;

		}
```

```
	}
	cout << "Максимальный размер достигнут" << endl;
	return l1;

}

bool operator !=(List& l6, List& l7) {

	Element* p1Temp = l6.pHead;
	Element* p2Temp = l7.pHead;
	while (p1Temp != NULL) {
		if (p1Temp->data != p2Temp->data) {
			p1Temp = p1Temp->next;
			p2Temp = p2Temp->next;
		}
		else {
			return false;
		}
	}
	return true;
}


int main(void) {
	setlocale(0, "");
	char x;
	List a, b;
	cout <<endl<< "Введите элемент первого списка: " << endl;
	int i = 0; bool add = true;
	while (add) {
		i++;
		cout << "Введите элемент: ";
		cin >> x;
		a.addToList(x);
		if (i < MAX) {
			cout << "Добавить еще?(y/n) ";
			cin >> x;
			if (x != 'y')
				add = false;
		}
		else {
			cout << "Максимальный размер достигнут" << endl;
			add = false;
		}
	}
	cout <<endl<< "Введите элемент второго списка: " << endl;
	i = 0; add = true;
	while (add) {
		i++;
		cout << "Введите элемент: ";
		cin >> x;
		b.addToList(x);
		if (i < MAX) {
			cout << "Добавить еще?(y/n) ";
			cin >> x;
			if (x != 'y')
				add = false;
		}
		else {
			cout << "Максимальный размер достигнут" << endl;
			add = false;
		}
	}
	cout << endl << "ПЕРВЫЙ ЛИСТ " << endl;
	a.printList();
```

```
        cout << endl << "ВТОРОЙ ЛИСТ " << endl;
        b.printList();
        cout <<endl<< "!ДОСТУП К ЭЛЕМЕНТУ! третий элемент первого списка: ";
        cout << a[2]->data << endl;
        if (a != b) {
                cout << "СПИСКИ НЕ ЯВЛЯЮТСЯ РАВНЫМИ" << endl;
        }
        else {
                cout << "СПИСКИ РАВНЫ" << endl;
        }
        cout << "ОБЪЕДИНЕНИЕ СПИСКОВ " << endl;
        (a + b).printList();
}
```

Вывод:



Вывод: получила практические навыки создания абстрактных типов данных и перегрузки операций в языке C++.