Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Пермский национальный исследовательский политехнический университет»
Электротехнический факультет
Кафедра «Информационные технологии и автоматизированные системы»

ЛАБОРАТОРНАЯ РАБОТА №6

«АТД. Контейнеры»

Выполнила:
студентка группы ИВТ-23-2б
Соловьева Екатерина
Александровна
Проверила:
доцент кафедры ИТАС
О. А. Полякова

2024 г.

Постановка задачи

1. Определить класс-контейнер.
2. Реализовать конструкторы, деструктор, операции ввода-вывода, операцию присваивания.
3. Перегрузить операции, указанные в варианте.
4. Реализовать класс-итератор. Реализовать с его помощью операции последовательного доступа.
5. Написать тестирующую программу, иллюстрирующую выполнение операций.

11 Вариант:

Класс- контейнер СПИСОК с ключевыми значениями типа int. Реализовать операции: []– доступа по индексу;
int() – определение размера списка;
+ вектор – сложение элементов списков a[i]+b[i];
- n - переход влево к элементу с номером n (с помощью класса-итератора).

Код программы на C++

Iterator.h

```
#pragma once
#include <iostream>

using namespace std;

class Iterator {
private:
	friend class List;
	int* elem;
public:
	Iterator();
	void operator++();
	void operator--();
	int& operator *() const;
	bool operator!=(const Iterator& it);
};
```

List.h

```
#pragma once
#include <iostream>
#include "Iterator.h"

using namespace std;

class List {
public:
	List(int s, int k = 0);
	List(const List& a);
	~List();
	List& operator =(const List& a);
```

```
        int& operator[] (int index);
        List operator +(const int k);
        int operator()();

        friend ostream& operator <<(ostream& out, const List&);
        friend istream& operator >> (istream& in, List& a);

        Iterator first() { return beg; }
        Iterator last() { return end; }

private:
        int size;
        int* data;
        Iterator beg, end;
};
```

Iterator.cpp

```
#include"Iterator.h"
#include <iostream>

using namespace std;
Iterator::Iterator() { elem = 0; }
void Iterator::operator++() { ++elem; }
void Iterator::operator--() { --elem; }
int& Iterator::operator *()const { return*elem; }
bool Iterator::operator!=(const Iterator& it) { return elem != it.elem; }
```

List.cpp

```
#include "List.h"
#include <iostream>

using namespace std;

List::List(int s, int k) {
        size = s;
        data = new int[size];
        for (int i = 0; i < size; i++) {
                data[i] = k;
        }
        beg.elem = &data[0];
        end.elem = &data[size];
}

List::List(const List& a) {
        size = a.size;
        data = new int[size];
        for (int i = 0; i < size; i++) {
                data[i] = a.data[i];
        }
        beg = a.beg;
        end = a.end;
}

List::~List() {
        delete[]data;
        data = 0;
}

List& List::operator =(const List& a) {
        if (this == &a) {
                return *this;
        }
        if (data != 0) {
                delete[]data;
        }
```

```
		data = new int[size];
		for (int i = 0; i < size; i++) {
			data[i] = a.data[i];
		}
		beg = a.beg;
		end = a.end;
		return *this;
}

int& List::operator[](int index) {
		if (index < size) {
			return data[index];
		}
		else {
			cout << "\nError (index > size)\n";
		}
}

List List::operator+(const int k) {
		List temp(size);
		for (int i = 0; i < size; i++) {
			temp.data[i] += data[i] + k;
		}
		return temp;
}

int List::operator() () {
		return size;
}

ostream& operator << (ostream& out, const List& a) {
		for (int i = 0; i < a.size; ++i) {
			cout << "OUT: " << i << " = ";
			out << a.data[i] << " ";
		}
		return out;
}

istream& operator >> (istream& in, List& a) {
		for (int i = 0; i < a.size; i++) {
			cout << "IN: " << i << " = ";
			in >> a.data[i];
		}
		return in;
}
```

Main.cpp

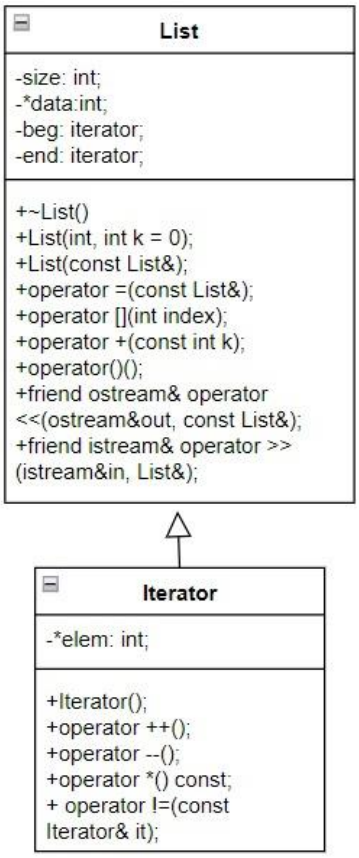
```
#include <iostream>
#include "List.h"
using namespace std;

int main() {
		List a(5);
		cout << a << "\n";

		cout << "Enter size A-list";
		cin >> a;
		cout << "A-list: " << a << "\n";

		a[2] = 100;
		cout << "A-list now: " << a << endl;

		List b(10);
		cout << "B-list: " << b << endl;
```

```
        b = a;
        cout << "B-list now: " << b << endl;

        List c(10);
        c = b + 100;

        cout << "C-list: " << c << endl;
        cout << "The length of A-list: " << a() << endl;

        /*****for testing*****/
        cout << *(a.first()) << endl;
        Iterator i = a.first();
        ++i;
        cout << *i << endl;
        for (i = a.first(); i != a.last(); ++i) {
                cout << *i << endl;
        }

        return 0;
}
```

Вывод:

```
OUT: 0 = 0 OUT: 1 = 0 OUT: 2 = 0 OUT: 3 = 0 OUT: 4 = 0
Enter size A-listIN: 0 = 4
IN: 1 = 3
IN: 2 = 2
IN: 3 = 1
IN: 4 = 5
A-list: OUT: 0 = 4 OUT: 1 = 3 OUT: 2 = 2 OUT: 3 = 1 OUT: 4 = 5
A-list now: OUT: 0 = 4 OUT: 1 = 3 OUT: 2 = 100 OUT: 3 = 1 OUT: 4 = 5
B-list: OUT: 0 = 0 OUT: 1 = 0 OUT: 2 = 0 OUT: 3 = 0 OUT: 4 = 0 OUT: 5 = 0 OUT: 6 = 0 OUT: 7 = 0 OUT: 8 = 0 OUT: 9 = 0
```

UML-диаграмма



Анализ результатов

Программа сработала корректно и вывела необходимые результаты.