Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

Пермский национальный исследовательский политехнический университет

Электротехнический факультет

Кафедра информационных технологий и автоматизированных систем

**ОТЧЕТ**

**о работе по информатике**

Семестр: 2

На тему: «ООП. АТД. Контейнеры»

Выполнил студент ИВТ-22-2б:

Афонин Артем Александрович

Проверил доцент кафедры ИТАС:

Полякова Ольга Андреевна

Пермь 2023

**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.

*Вариант 15:*

Класс- контейнер СПИСОК с ключевыми значениями типа int. Реализовать операции:

[] – доступа по индексу;

int() – определение размера списка;

\* вектор – умножение элементов списков a[i]\*b[i];

+n - переход вправо к элементу с номером n ( с помощью класса-итератора).

**Описание классов**

Класс Node (узел списка). Поля с доступом public: int data – поле данных целого типа, Node\* - указатель на следующий узел.

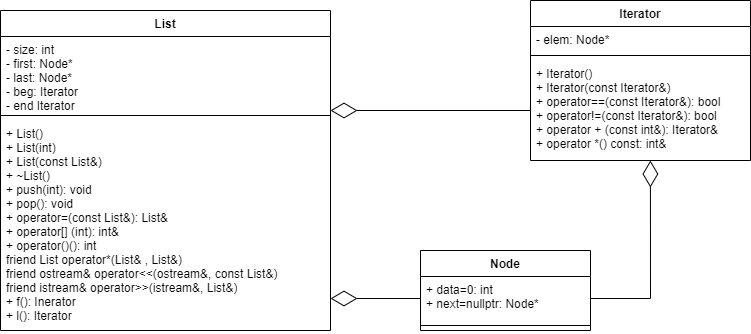
Класс List (список). Поля с доступом private: int size – размер списка, Node\*first – указатель на первый элемент списка, Node\* last – указатель на последний элемент списка; Iterator beg – указатель на первый элемент вектора, Iterator end – указатель на элемент, следующий за последним.

Методы с доступом public: 3 вида конструкторов и деструктор; методы push() и pop() для добавления и удаления элементов; перегрузка операторов для методов присваивания, доступа по индексу, определения размера списка. Дружественные функции для ввода/вывода списка и для умножения элементов списка; методы возвращающие указатели на первый элемент и на элемент, следующий за последним.

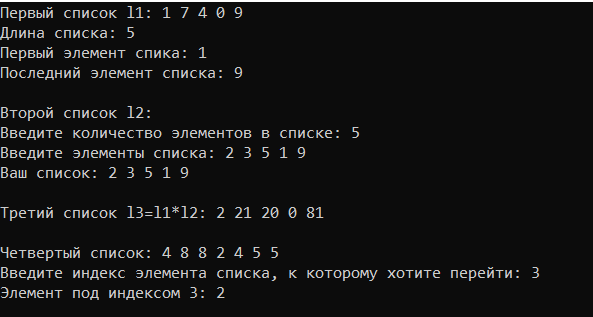
Класс Iterator. Поля с доступом private: Node\* elem – указатель на узел списка.

Методы с доступом public: конструкторы; перегрузка операторов для методов сравнения, перехода вправо к элементу с номером n, разыменовывания.

Также описание классов представлено на UML-диаграмме:



**Результат работы программы**



**Контрольные вопросы**

*1. Что такое абстрактный тип данных? Привести примеры АТД.*

АТД − тип данных, определяемый только через операции, которые могут выполняться над соответствующими объектами безотносительно к способу представления этих объектов.

template <typename T>

T& Stack<T>::push() { //операция добавления элементов в АТД типа стек

return head->data;

}

*2. Привести примеры абстракции через параметризацию.*

template <typename T>

class node {

T data;

node\* next, \* prev;

}

*3. Привести примеры абстракции через спецификацию.*

class complex {

double re, im;

friend complex operator+(complex, complex);

public:

complex() { re = im = 0; }

complex(double r) { re = r; im = 0; }

complex(double r, double i) { re = r; im = i; }

};

complex operator+(complex a1, complex a2) {

return complex(a1.re + a2.re, a1.im + a2.im);

}

Для поддержки смешанной арифметики комплексных и действительных чисел (Re x, In y) + double A требуется специфицировать перегруженную функцию.

complex operator+(complex, complex);

complex operator+(complex, double);

complex operator+(double, complex);

Абстракция через спецификацию достигается за счет того, что операции представляются как часть типа (абстракция вычислений, перегрузка операторов).

*4. Что такое контейнер? Привести примеры.*

Контейнер – это объект, содержащий другие однотипные объекты. Примеры: двусвязный список, массив, стек, множество.

class queue() {

int size;

node\* head, \* tail;

};

int main() {

queue\* q; //контейнер-очередь

float a = 7.5;

q->head = a; //головой очереди q является float переменная

queue\* c;

c->tail = q; //хвостом очереди c является контейнер queue

return 0;

}

*5. Какие группы операций выделяют в контейнерах?*

* Операции доступа к элементам, которые обеспечивают и операцию замены значений элементов;
* Операции добавления и удаления элементов или групп элементов;
* Операции поиска элементов и групп элементов;
* Операции объединения контейнеров.

*6. Какие виды доступа к элементам контейнера существуют? Привести примеры.*

Виды доступа: последовательный, прямой и ассоциативный.

Последовательный доступ – перемещение от элемента к элементу. Пример: a.next.

Прямой доступ – это доступ по индексу. Например, a[10].

Ассоциативный доступ также выполняется по индексу, но индексом будет являться не номер элемента, а его содержимое. Пример: a[“word”].

*7. Что такое итератор?*

Итератор – это объект, который обеспечивает последовательный доступ к элементам контейнера.

*8. Каким образом может быть реализован итератор?*

Итератор реализуется как класс, который имеет такой же интерфейс, как и указатель для совместимости с массивами.

*9. Каким образом можно организовать объединение контейнеров?*

Наиболее часто используется операция объединения двух контейнеров с получением нового контейнера. Она может быть реализована в разных вариантах:

* Простое сцепление двух контейнеров: в новый контейнер попадают сначала элементы первого контейнера, потом второго, операция не коммутативна.
* Объединение упорядоченных контейнеров, новый контейнер тоже будет упорядочен, операция коммутативна.
* Объединение контейнеров как объединение множеств, в новый контейнер попадают только те элементы, которые есть хотя бы в одном контейнере, операция коммутативна.
* Объединение контейнеров как пересечение множеств, в новый контейнер попадают только те элементы, которые есть в обоих контейнерах, операция коммутативна.

*10. Какой доступ к элементам предоставляет контейнер, состоящий из элементов «ключ-значение»?*

Ассоциативный доступ.

*11. Как называется контейнер, в котором вставка и удаление элементов выполняется на одном конце контейнера?*

Стек.

*12. Какой из объектов (a,b,c,d) является контейнером?*

a. int mas=10;

b. 2. int mas;

c. 3. struct {char name[30]; int age;} mas;

d. 4. int mas[100];

Ответ: d

*13. Какой из объектов (a,b,c,d) не является контейнером?*

a. int a[]={1,2,3,4,5};

b. 2. int mas[30];

c. 3. struct {char name[30]; int age;} mas[30];

d. 4. int mas;

Ответ: d

*14. Контейнер реализован как динамический массив, в нем определена операция доступ по индексу. Каким будет доступ к элементам контейнера?*

Прямой доступ.

*15. Контейнер реализован как линейный список. Каким будет доступ к элементам контейнера?*

Последовательный доступ.