Проект "Йерархия от контейнери"

Програмата представлява хетерогенен свързан списък от състоящ се от **n** контейнера. Числото **n** се въвежда от потребителя, както и вида на контейнера и числата, който трябва да съдържа след което се записват във файл, който се състои от 2*n редове, горния ред показва вида на стека , а непосредствено под него са числата, които се съдържат в него.Всеки контейнер може да е шаблон на двусвързан списък, стек или опашка и притежава операция bool member(T const& x), която проверява дали даден елемент **x** се среща в контейнера.

Библиотеките(или помощните cpp файлове), които съм ползвала са :

#include <iostream> - за стандартния вход/изход

#include <string> - за представянето на вида на контейнера

#include<fstream> - за четене/писане във файл

#include<sstream> - за превръщането на прочетен ред от файл като стринг в ред от числа

#include<queue> - използвала съм <priority queue> като помощна структура за сортиране на контейнерите

#include "lstack.cpp" – версията на свързан стек от лекции

#include "lqueue.cpp" – версията на свързана опашка от лекции

#include "linked_list.cpp" – версията на двойно свързан лист от лекции, който е нужен за представеното на самия хетерогенен списък

#include "double_linked_list.cpp" – версията на свързан стек от лекции

В проекта съм използвала 5 основни класа :

class Object – интерфейс за имплементирането на обектите в хетерогенния контейнер.

class StackObject : public Object<T>, private LinkedStack<T> - представя обекти, който са едновременно стекове и обекти от хетерогенния контейнер

class QueueObject : public Object<T>, private LinkedQueue<T> - представя обекти, който са едновременно опашки и обекти от хетерогенния контейнер

class DoubleListObject : public Object<T>, private DoubleLinkedList<T> - представя обекти, който са едновременно списъци и обекти от хетерогенния контейнер

class QueueStackList : public LinkedList<Object<int>*> - самия хетерогенен контейнер;

Следва описание на основните функции за всеки класа

1. class Object

- **virtual bool insert(T const&) =0** – чисто виртуална функция, служеща за добавяне на елемент в съответния контейнер
- **virtual bool remove(T&) = 0** - чисто виртуална функция, служеща за премахване на елемент от съответния контейнер
- **virtual bool member(T const&) = 0** - чисто виртуална функция, служеща за проверка дали даден елемент се среща в съответния контейнер
- **virtual int l_size() = 0** – връща броя на елементите в съответния контейнер
- **virtual void sort() = 0** – сортира съответния контейнер
- **virtual void obj_filter(Condition) = 0** – филтрира съответния контейнер според предварително зададено условие Condition
- **virtual bool special_condition(Condition) = 0** – проверява дали някой от елементите на съответния контейнер отговаря на предварително зададено условие Condition
- **virtual void print(ostream& os) const = 0** – принтира върху конзолата съответния контейнер

Класовете : QueueObject , StackObject , DoubleListObject представляват имплементация на класа Object.

2.class QueueStackList

- **~QueueStackList()** - деструктор
- **void read_from_file()** – четене от файл и вкарване на информацията в хетерогенния контейнера
- **bool if_in_conteiner(int)** – проверка дали даден елемент се среща в някой от контейнерите
- **void all_filter(Condition)** – филтрира всички елементи на хетерогенния контейнер по предварително зададено условие
- **void l_insert(int)** – добавяне на елемент в контейнер, с най-малък размер
- **void l_sort()** - сортиране на всички контейнери
- **void from_l_to_txt()** – вкарване на информацията от хетерогенния списък в .txt файл
- **void print()** – принтиране на хетерогенния списък