Курс: Объектно-ориентированное программирование на C++

Встреча №11

ТЕМА: ШАБЛОНЫ ФУНКЦИЙ

Задания для самостоятельной работы:

Задание №1

Написать шаблон функции для поиска среднего арифметического значений массива.

Задание №2

Написать перегруженные шаблоны функций для нахождения корней линейного (a*x + b = 0) и квадратного (a*x2+b*x + c = 0) уравнений.

Замечание: *в функции передаются коэффициенты уравнений.*

Задание №3

Напишите шаблон функции, которая возвращает максимум из двух переданных параметров

Задание №4

Напишите шаблон функции, которая возвращает минимум из двух переданных параметров.

Курс: Объектно-ориентированное программирование на C++

Встреча №12

ТЕМА: ШАБЛОНЫ КЛАССОВ

Задания для самостоятельной работы:

Задание 1

Создать шаблонный класс-контейнер ***Array,*** который представляет собой массив, позволяющий хранить объекты заданного типа.

Класс должен реализовывать следующие функции:

- **GetSize** – получение размера массива (количество элементов, под которые выделена память);
- **SetSize(int size, int grow = 1)** – установка размера массива (если параметр ***size*** больше предыдущего размера массива, то выделяется дополнительный блок памяти, если нет, то «лишние» элементы теряются и память освобождается); параметр ***grow*** определяет для какого количества элементов необходимо выделить память, если количество элементов превосходит текущий размер массива. Например, SetSize(5, 5); означает, что при добавлении 6-го элемента размер массива становится равным 10, при добавлении 11-го - 15 и т. д.;
- **GetUpperBound** - получение последнего допустимого индекса в массиве. Например, если при размере массива 10, вы добавляете в него 4 элемента, то функция вернет 3;
- **IsEmpty** - массив пуст?;

Лабораторная работа №12

- FreeExtra - удалить «лишнюю» память (выше последнего допустимого индекса);
- **RemoveAll** – удалить все;
- **GetAt** -получение определенного элемента (по индексу);
- **SetAt** – установка нового значения для определенного элемента (индекс элемента должен быть меньше текущего размера массива);
- **operator** [] – для реализации двух предыдущих функций;
- **Add** – добавление элемента в массив (при необходимости массив увеличивается на значение ***grow*** функции ***SetSize***);
- **Append** – «сложение» двух массивов;
- **operator =;**
- **GetData** – получения адреса массива с данными;
- **InsertAt** – вставка элемента(-ов) в заданную позицию;
- **RemoveAt** – удаление элемента(-ов) с заданной позиции.

Курс: Объектно-ориентированное программирование на C++

Встреча №13

ТЕМА: ДИНАМИЧЕСКИЕ СТРУКТУРЫ ДАННЫХ – СТЕК

Задания для самостоятельной работы:

Задание №1

Реализуйте класс стека для работы с символами (символьный стек). Стек должен иметь фиксированный размер. Также реализуйте набор операций для работы со стеком: помещение символа в стек, выталкивание символа из стека, подсчет количества символов в стеке, проверку пустой ли стек, проверку полный ли стек, очистку стека, получение без выталкивания верхнего символа в стеке.

Задание №2

Измените стек из первого задания со статического типа на динамический (при нехватке свободного места нужно изменить размер внутреннего массива без потери данных).

Лабораторная работа №14

Курс: Объектно-ориентированное программирование на C++

Встреча №14

ТЕМА: ДИНАМИЧЕСКИЕ СТРУКТУРЫ ДАННЫХ – ОЧЕРЕДЬ, ОЧЕРЕДЬ С ПРИОРИТЕТАМИ

Задания для самостоятельной работы:

Задание №1

Создайте шаблонный класс обычной очереди для работы с целыми значениями. Требуется создать реализации для типичных операций над элементами:

- **IsEmpty** – проверка очереди на пустоту
- **IsFull** – проверка очереди на заполнение
- **Enqueue** – добавление элемента в очередь
- **Dequeue** – удаление элемента из очереди
- **Show** – отображение всех элементов очереди на экран

Задание №2

Создайте класс очереди с приоритетами для работы с целыми значениями. Требуется создать реализации для типичных операций над элементами очереди:

- **IsEmpty** – проверка очереди на пустоту
- **IsFull** – проверка очереди на заполнение
- **InsertWithPriority** – добавление элемента с приоритетом в очередь
- **PullHighestPriorityElement** – удаление элемента с самым высоким приоритетом из очереди

- **Peek** – возврат самого большого по приоритету элемента. Обращаем ваше внимание, что элемент не удаляется из очереди.
- **Show** – отображение всех элементов очереди на экран. При показе элемента также необходимо отображать приоритет.

Задание №3

Измените класс из задания 2 на шаблонный класс.