Московский авиационный институт

(национально исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра прикладной математики и информатики

Лаборатория №2 по курсу «Объектно-ориентируемое программирование»

Студент: Рождественских Аднрей Сергеевич

Преподаватель: А. В. Поповкин

Группа: М80-206Б

Вариант: 15

Дата:

Оценка:

Подпись:

Москва, 2018.

1 Цель работы

• Закрепление навыков работы с классами.

• Создание простых динамических структур данных.

• Работа с объектами, передаваемыми «по значению».

2 Задача

Необходимо спроектировать и запрограммировать на языке С++ класс-контейнер первого уровная, содержащий одну фигуру, согласно варианту задания.

Классы должны удовлетворять следующим правилам:

• Требования к классу фигуры аналогичны требованиям из лабораторной работы 1.

• Классы фигур должны иметь переопределенный оператор вывода в поток std::ostream(«). Оператор должен распечатывать параметры фигуры.

• Классы фигур должны иметь переопределенный оператор ввода фигуры из потока std::istream(»). Оператор должен вводить параметры фигуры.

• Классы фигур должны иметь операторы копирования (=).

• Классы фигур должны иметь операторы сравнения с такими же фигурами (==).

• Класс-контейнер должен содержать объекты фигур "по значению"(не по ссылке).

• Класс-контейнер должен иметь метод по добавлению фигуры в контейнер.

• Класс-контейнер должен иметь методы по получению фигуры из контейнера.

• Класс-контейнер должен иметь метод по удалению фигуры из контейнера.

• Класс-контейнер должен иметь перегруженный оператор по выводу контейнера в поток std::ostream(«).

• Класс-контейнер должен иметь деструктор, удаляющий все элементы контейнера.

• Классы должны быть расположены в раздельных файлах: отдельно заголовки (.h), отдельно описание методов (.cpp).

Фигура: шестиугольник. Контейнер: Бинарное дерево.

3 Описание

Динамические структуры данных используются в тех случаях, когда мы заранее не знаем, сколько памяти необходимо выделить для нашей программы – это выясняется только в процессе работы. В общем случае эта структура представляет собой отдельные элементы, связанные между собой с помощью ссылок. Каждый элемент состоит из двух областей памяти: поля данных и ссылок. Ссылки – это адреса других узлов того же типа, с которыми данный элемент логически связан. При добавлении нового элемента в такую структуру выделяется новый блок памяти и устанавливаются связи этого элемента с уже существующими.

Параметры в функцию могут передаваться одним из следующих способов: по значению и по ссылке. При передаче аргументов по значению компилятор создает временную копию объекта, который должен быть передан, и размещает его в области стековой памяти, предназначенной для хранения локальных объектов. Вызываемая функция оперирует именно с этой копией, не оказывая влияния на оригинал объекта. Прототипы функций, принимающих аргументы по значению, предусматривают в качестве параметров указание типа объекта, а не его адреса. Если же необходимо, чтобы функция модифицировала оригинал объекта, используется передача параметров по ссылке. При этом в функцию передается не сам объект, а только его адрес. Таким образом, все модификации в теле функции переданных ей по ссылке аргументов воздействуют на объект. Использование передачи адреса объекта весьма эффективный способ работы с большим числом данных. Кроме того, так как передается адрес, а не сам объект, существенно экономится стековая память.

4 Изменения

Добавились файлы TbinaryTree.h TbinaryTree.cpp TbinaryTreeItem.cpp TbinaryTreeItem.h

5 Выводы

В этой работе было необходимо реализовать собственную структуру данных. В моем случае ей являлся массив. В него можно добавлять произвольное количество элементов в конец, печатать содержимое, брать элемент по индексу, удалять элемент по индексу. Несмотря на то, что все основные структуры уже созданы до нас, каждый программист должен уметь реализовать их.