**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 6**

Тема: Основы работы с коллекциями: аллокаторы

Студент: Курносов М. Ю.

Группа: 80-208

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2020

**Постановка задачи**

**Вариант 25**

Разработать шаблоны классов согласно варианту задания. Параметром шаблона должен являться скалярный тип данных задающий тип данных для оси координат. Классы должны иметь публичные поля. Фигуры являются фигурами вращения, т.е. равносторонними (кроме трапеции и прямоугольника). Для хранения координат фигур необходимо использовать шаблон std::pair.

Например:

template <class T>

struct Square{

using vertex\_t = std::pair<T,T>;

vertex\_t a,b,c,d;

};

Создать шаблон динамической коллекцию, согласно варианту задания:

1. Коллекция должна быть реализована с помощью умных указателей (std::shared\_ptr, std::weak\_ptr). Опционально использование std::unique\_ptr;

2. В качестве параметра шаблона коллекция должна принимать тип данных;

3. Коллекция должна содержать метод доступа:

o Стек – pop, push, top;

o Очередь – pop, push, top;

o Список, Динамический массив – доступ к элементу по оператору [];

4. Реализовать аллокатор, который выделяет фиксированный размер памяти (количество блоков памяти – является параметром шаблона аллокатора). Внутри аллокатор должен хранить указатель на используемый блок памяти и динамическую коллекцию указателей на свободные блоки. Динамическая коллекция должна соответствовать варианту задания (Динамический массив, Список, Стек, Очередь);

5. Коллекция должна использовать аллокатор для выделения и освобождения памяти для своих элементов.

6. Аллокатор должен быть совместим с контейнерами std::map и std::list (опционально – vector).

7. Реализовать программу, которая:

o Позволяет вводить с клавиатуры фигуры (с типом int в качестве параметра шаблона фигуры) и добавлять в коллекцию использующую аллокатор;

o Позволяет удалять элемент из коллекции по номеру элемента;

o Выводит на экран введенные фигуры c помощью std::for\_each;

Вариант:

|  |  |  |  |
| --- | --- | --- | --- |
| 25. | Треугольник | Динамический массив | Динамический массив |

**Описание программы**

Программа состоит из трёх файлов: main.cpp, triangle.hpp и vector.hpp. В файле triangle.hpp находится шаблон треугольника (фигура взята в соответствии с вариантом), который хранится в виде трёх вершин, заданных с помощью std::pair. По варианту, наша коллекция представляет из себя очередь, поэтому в файле vector.hpp с помощью умных указателей. В данном файле вектор и итератор определены в виде классов (TVector и Iterator), содержащие соответствующие методы взаимодействия. Iterator находится внутри класса векторa, где и определены основные методы: вставка, удаление по итератору, PushEnd, PopEnd и другие. Вектор хранит в себе объекты шаблонного класса Треугольников. Также подключен аллокатор, который вызывается при создании вектора.

vector.hpp - файл для шаблона класса аллокатора, который выделяет память фиксированного размера с помощью вектора vector, описанного в этом же файле. Содержит методы аллоцирования и деаллоцирования памяти.

Основная работа программы происходит в main.cpp. При запуске программы создается динамический массив с 3 пустыми ячейками память на которую выделяется аллокатором, и пользователю предоставляется доступ к меню, с помощью которого он может добавлять, удалять треугольники в векторе, печатать вектор, и сравнивать введенную площадь с площадью других треугольников. Программа принимает на вход только значения типа int (согласно заданию). Программа способна уведомлять пользователя о неправильном вводе и отменить текущую операцию. Добавление и удаление осуществляется в виде работающих по индексу функций. Также пользователь может посчитать количество фигур с площадью меньшей, чем введённая. Для этого он выбирает соответствующее действие и вводит площадь для сравнения. Данный метод работает с помощью std::count\_if. Программа завершает свою работу только после того, как этого захотел сам пользователь (выбрал соответствующее действие в меню выбора(под номером 0)).

**Набор тестов**

|  |  |  |
| --- | --- | --- |
| For file test\_01.txt: | For file test\_02.txt: | For file test\_03.txt: |
| 1  2 5  3 6  3  1  5 6  7 8  1  8 9  10 11  1  100 0  0 100  3  4  100000  4  10 ф | 1  2 5  3 6  3  1  5 6  7 8  1  8 9  10 11  1  100 0  0 100  3  2  22  3  2  2  3 | 1  2 5  3 6  3  1  5 6  7 8  1  8 9  10 11  1  100 0  0 100  3  4  100000  4  10  2  22  3  2  2  3  9  0 |

Пояснение к тестам:

В 1 тесте мы тестируем вставку элементов и функцию подсчёта фигур с площадью меньше введённой, а так же введение буквы в программу.

Во 2 тесте мы проверяем удаление фигур, а так же ввод через файл без терминатора программы(не вводим 0 по завершении).

В 3 тесте мы проверяем всё ещё раз и вводим цифру которая не предусмотрена программой(программа реагирует и выводит справку).

**Результат выполнения тестов**

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_01.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Похоже введена буква, ну в другой раз не промахнётесь по клавиатуре ;-)

-->1853121906<--

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_01.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Похоже введена буква, ну в другой раз не промахнётесь по клавиатуре ;-)

-->1853121906<--

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_02.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите номер удаляемой фигуры

Вектор пуст, удалять - нечего.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите номер удаляемой фигуры

Фигура удалена

Текущий размер хранилища :Size = 3

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №3

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Похоже введена буква, ну в другой раз не промахнётесь по клавиатуре ;-)

-->1853121906<--

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_03.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Введите номер удаляемой фигуры

Вектор пуст, удалять - нечего.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите номер удаляемой фигуры

Фигура удалена

Текущий размер хранилища :Size = 3

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №3

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Выход из программы...

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_01.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Похоже введена буква, ну в другой раз не промахнётесь по клавиатуре ;-)

-->1853121906<--

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_01.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Похоже введена буква, ну в другой раз не промахнётесь по клавиатуре ;-)

-->1853121906<--

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_01.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Похоже введена буква, ну в другой раз не промахнётесь по клавиатуре ;-)

-->1853121906<--

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# g++ -o p main.cpp

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_01.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Похоже не цифра введена, ну в другой раз не промахнётесь по клавиатуре ;-)

--><--

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_01.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Похоже не цифра введена, ну в другой раз не промахнётесь по клавиатуре ;-)

--><--

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_01.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Похоже не цифра введена, ну в другой раз не промахнётесь по клавиатуре ;-)

--><--

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_01.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Похоже не цифра введена, ну в другой раз не промахнётесь по клавиатуре ;-)

--><--

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# g++ -o p main.cpp

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_01.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Выход из программы...

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_01.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Похоже не цифра введена, ну в другой раз не промахнётесь по клавиатуре ;-)

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_01.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Похоже не цифра введена, ну в другой раз не промахнётесь по клавиатуре ;-)

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_02.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите номер удаляемой фигуры

Вектор пуст, удалять - нечего.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите номер удаляемой фигуры

Фигура удалена

Текущий размер хранилища :Size = 3

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №3

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Выход из программы...

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_03.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Введите номер удаляемой фигуры

Вектор пуст, удалять - нечего.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите номер удаляемой фигуры

Фигура удалена

Текущий размер хранилища :Size = 3

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №3

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Выход из программы...

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./see test\*

.see.output .vscode allocator.hpp allocvector.hpp main.cpp p see test\_01.txt test\_02.txt test\_03.txt triangle.hpp vector.hpp

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_01.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки. ooproot@g8-RM:/mnt/c/Users/g7/De

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Похоже не цифра введена, ну в другой раз не промахнётесь по клавиатуре ;-)

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_02.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите номер удаляемой фигуры

Вектор пуст, удалять - нечего.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите номер удаляемой фигуры

Фигура удалена

Текущий размер хранилища :Size = 3

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №3

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Выход из программы...

root@g8-RM:/mnt/c/Users/g7/Desktop/коды/2sem/oop/6laboop# ./p < test\_03.txt

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки.

Элемент вставлен.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите площадь больше площадей только тех фигур , которые надо вывести

4

Введите площадь больше площадей только тех фигур , которые надо вывести

3

Введите номер удаляемой фигуры

Вектор пуст, удалять - нечего.

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 5 y = 6

Вершина 2 x = 7 y = 8

Вершина 3 x = 4.26873 y = 8.73495

Фигура №3

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №4

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Введите номер удаляемой фигуры

Фигура удалена

Текущий размер хранилища :Size = 3

Фигура №1

Вершина 1 x = 2 y = 5

Вершина 2 x = 3 y = 6

Вершина 3 x = 1.63436 y = 6.36748

Фигура №2

Вершина 1 x = 8 y = 9

Вершина 2 x = 10 y = 11

Вершина 3 x = 7.26873 y = 11.735

Фигура №3

Вершина 1 x = 100 y = 0

Вершина 2 x = 0 y = 100

Вершина 3 x = -36.7476 y = -36.5636

Для выполнения операции введите соответствующую цифру:

1 - Добавить фигуру

2 - Удалить фигуру

3 - Вывести все фигуры на экран

4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран

0 - Выйти из программы

Выход из программы...

**Листинг программы**

([git@github.com:Arhangel333/oop\_exercise\_06.git](mailto:git@github.com:Arhangel333/oop_exercise_06.git))

**For file main.cpp:**

#include <iostream>

#include <iterator>

#include <algorithm>

#include <memory>

#include <cmath>

#include "vector.hpp"

#include "triangle.hpp"

void help()

{

printf("Для выполнения операции введите соответствующую цифру:\n ");

printf("\t 1 - Добавить фигуру\n ");

printf("\t 2 - Удалить фигуру\n ");

printf("\t 3 - Вывести все фигуры на экран\n ");

printf("\t 4 - Вывести количество всех фигур, у которых площадь меньше, заданной, на экран\n ");

printf("\t 0 - Выйти из программы\n ");

}

int main()

{

int i = 1;

char enter = -1;

TP xl, yl, xr, yr;

int c;

TVector<Triangle<TP>, Alloc<Triangle<TP>, 10000>> vec;

help();

std::cin >> enter;

while (isdigit(enter))

{

switch (enter - 48)

{

case 1:

{

cout << "Введите координаты двух вершин вашей фигуры в направлении против часовой стрелки." << endl;

std::cin >> xl >> yl >> xr >> yr;

/\* Triangle<TP>\* ptr = allocmain.allocate();

\*ptr = ; \*/

Triangle<TP> a(xl, yl, xr, yr);

vec.insert(vec.size, a);

//ptr = NULL;

//allocmain.deallocate(ptr);

cout << "Элемент вставлен." << endl;

break;

}

case 2:

{

cout << "Введите номер удаляемой фигуры " << endl;

int numb;

std::cin >> numb;

int err = (vec.Erase(numb - 1));

if (err < 0)

{

switch (err)

{

{

case -1:

{

printf("Вектор пуст, удалять - нечего.\n");

break;

} case -2:

{

printf("В векторе нет элемента под индексом %d\n", numb);

break;

}

default:

printf("Ошибка при удалении вектора.\n");

break;

}

}

break;

}

cout << "Фигура удалена\n Текущий размер хранилища :Size = " << vec.size << endl;

break;

}

case 3:

{ //вывод всех фигур

for\_each(vec.begin(), vec.end(), [&](Triangle<TP> &&tr) {

printf("Фигура №%d\n", i);

i++;

tr.Print();

});

i = 1;

break;

}

case 4:

{

TP square = 0;

printf("Введите площадь больше площадей только тех фигур , которые надо вывести\n");

std::cin >> square;

//вывод если прлощадь меньше заданной

int r = std::count\_if(vec.begin(), vec.end(), [&](Triangle<TP> &&tr) { return tr.Square() < square; });

cout << r << endl;

break;

}

case 0:

{

cout << "Выход из программы..." << endl;

return 0;

}

default:

{

help();

break;

}

}

enter = 48;

std::cin >> enter;

}

if(!(isdigit(enter))){

cout << "Похоже не цифра введена, ну в другой раз не промахнётесь по клавиатуре ;-) \n\n"<< endl;

}

return 0;

}

**For file vector.hpp:**

#include <stdexcept>

using namespace std;

typedef double TP;

template <class T, size\_t BLOCK\_SIZE>

class Alloc;

template <class T>

class vector;

template <class T, class Alloc>

class TVector

{

public:

const int TVECTOR\_EXTRA\_SIZE = 5;

int size;

int capacity;

T \*data;

static const size\_t memSize = 10000; //фиксированный размер аллокатора

//friend class Alloc<T,memSize>;

Alloc alloc; //экземпляр аллокатора

TVector()

{

this->size = 0;

this->capacity = 3;

this->data = alloc.Alloc::allocate(3);

}

TVector(const TVector<T, Alloc> &v)

{

this->size = v.size;

this->capacity = v.capacity;

this->data = alloc.Alloc::allocate(v.capacity);

for (int i = 0; i < this->size; i++)

{

this->data[i] = v.data[i];

}

}

TVector(int x)

{

this->size = 0;

this->capacity = x;

this->data = alloc.Alloc::allocate(x);

}

void PushEnd(T x)

{

if (this->size >= this->capacity)

{

int oldcap = this->capacity;

this->capacity += TVECTOR\_EXTRA\_SIZE;

T \*buf = alloc.Alloc::allocate(this->capacity);

for (int i = 0; i < oldcap; i++)

{

buf[i] = this->data[i];

}

alloc.Alloc::deallocate(this->data, this->size);

this->data = buf;

buf = 0;

}

this->data[this->size] = x;

this->size++;

}

void PopEnd()

{

if (this->size <= 0)

{

printf("TVector is empty.\n");

}

else

{

this->size--;

}

}

int Erase(int i)

{

if (this->size <= 0)

{

return -1;

}

else

{

if ((i < this->size) && (i >= 0))

{

this->size--;

for (i; i < this->size; i++)

{

//if(i != )

this->data[i] = this->data[i + 1];

}

}

else

{

return -1;

}

}

return 0;

}

void insert(int i, T tr)

{

if ((i <= this->size) && (i >= 0))

{

this->PushEnd(tr);

for (int j = this->size - 1; j > i; j--)

{

this->data[j].Print();

this->data[j] = this->data[j - 1];

}

this->data[i] = tr;

}

}

class iterator

{

public:

int index = -1;

TVector<T, Alloc> \*vect;

//std::iterator\_traits<TVector<T, Alloc>::iterator> difference\_type = int;

using difference\_type = ptrdiff\_t;

using value\_type = T;

using reference = T &;

using pointer = T \*;

using iterator\_category = std::forward\_iterator\_tag;

std::shared\_ptr<T> ptr;

iterator()

{

index = 0;

}

iterator(const iterator &it)

{

this->vect = it.vect;

this->ptr = it.ptr;

this->index = it.index;

}

iterator(int ind, TVector<T, Alloc> \*vect)

{

this->ptr = std::make\_shared<T>((vect->data[ind]));

this->index = ind;

this->vect = vect;

}

iterator operator++()

{

//cout<<this<<" "<<this->index<<" "<<this->vect->size<<endl;

if ((this->index + 1) > this->vect->size)

{

//printf("Iterator cannot be incremented past the end of range.%d %d\n", this->index, this->vect->size);

throw std::out\_of\_range("Iterator cannot be incremented past the end of range.");

return \*this;

}

this->index++;

this->ptr = std::make\_shared<T>(this->vect->data[this->index]);

return \*this;

}

iterator &operator++(int)

{

if ((this->index + 1) > (this->vect->size))

{

//printf("Iterator cannot be incremented past the end of range.\n");

throw std::out\_of\_range("Iterator cannot be incremented past the end of range.");

return \*this;

}

this->index++;

this->ptr = std::make\_shared<T>(this->vect->data[this->index]);

return \*this;

}

iterator operator=(iterator x)

{

//printf("=\n");

this->ptr = x.ptr;

return \*this;

}

bool operator!=(const iterator &x)

{ //printf("!=\n");

if (index != x.index)

return true;

//printf("false\n ");

/\* if(ptr != x.ptr) return true;

printf("\t !=\n "); \*/

return false;

}

bool operator>=(iterator x)

{

return this->ptr >= x.ptr;

}

int operator-(iterator x)

{

//printf("-\n");

return this->index - x.index;

}

bool operator==(const iterator &x)

{

//printf("\t ==\n ");

if (index == x.index)

return true;

if (ptr == x.ptr)

return true;

return false;

}

T operator\*()

{

return \*(this->ptr);

}

T \*operator->()

{

return this->ptr;

}

~iterator() {}

};

iterator begin()

{

TVector<T, Alloc>::iterator it(0, this);

return it;

}

iterator end()

{

TVector<T, Alloc>::iterator it(this->size, this);

return it;

}

void Printing()

{

printf(">>>>>>>>\n");

for (int i = 0; i < this->size; i++)

{

cout << this->data[i] << " ";

}

cout << endl;

printf(">>>>>>>>size %d | cap %d\n", this->size, this->capacity);

}

void Show()

{

printf(">>>>>>>>\n");

for (int i = 0; i < this->capacity; i++)

{

cout << this->data[i] << " ";

}

cout << endl;

printf(">>>>>>>>size %d | cap %d\n", this->size, this->capacity);

}

~TVector()

{

this->size = 0;

this->capacity = 0;

alloc.Alloc::deallocate(this->data, this->size);

}

T operator[](int i)

{

return this->data[i];

}

T \*operator++(int x)

{

return this->data[x + 1];

}

T \*operator--(int x)

{

return this->data[x - 1];

}

void operator=(TVector<T, Alloc> vec)

{

for (int i = 0; i < this->size; i++)

{

this->data[i] = vec.data[i];

}

}

};

template <class T>

class vector

{

public:

int size;

int capacity;

T \*data;

const int VECTOR\_EXTRA\_SIZE = 10;

// friend class TVector<T, Alloc<T, memSize>>;

vector()

{

this->size = 0;

this->capacity = 3;

this->data = new T[3];

}

vector(const vector<T> &v)

{

this->size = v.size;

this->capacity = v.capacity;

this->data = new T[v.capacity];

for (int i = 0; i < this->size; i++)

{

this->data[i] = v.data[i];

}

}

vector(int x)

{

this->size = 0;

this->capacity = x;

this->data = new T[x];

}

void PushEnd(T x)

{

if (this->size >= this->capacity)

{

int oldcap = this->capacity;

this->capacity += VECTOR\_EXTRA\_SIZE;

T \*buf = new T[this->capacity];

for (int i = 0; i < oldcap; i++)

{

buf[i] = this->data[i];

}

delete[] this->data;

this->data = buf;

buf = 0;

}

this->data[this->size] = x;

this->size++;

}

bool Empty()

{

if (this->size == 0)

{

return true;

}

return false;

}

T PopEnd()

{

if (this->size <= 0)

{

printf("Vector is empty.\n");

return -1;

}

else

{

this->size--;

}

return this->data[this->size];

}

void Erase(int i)

{

if (this->size <= 0)

{

printf("Vector is empty.\n");

}

else

{

if ((i < this->size) && (i >= 0))

{

this->size--;

for (i; i < this->size; i++)

{

//if(i != )

this->data[i] = this->data[i + 1];

}

}

else

{

printf("В векторе нет элемента под индексом %d\n", i);

}

}

}

void insert(int i, T tr)

{

if ((i <= this->size) && (i >= 0))

{

this->PushEnd(tr);

for (int j = this->size - 1; j > i; j--)

{

this->data[j].Print();

this->data[j] = this->data[j - 1];

}

this->data[i] = tr;

}

}

class iterator

{

public:

int index = -1;

vector<T> \*vect;

//std::iterator\_traits<vector<T>::iterator> difference\_type = int;

using difference\_type = ptrdiff\_t;

using value\_type = T;

using reference = T &;

using pointer = T \*;

using iterator\_category = std::forward\_iterator\_tag;

std::shared\_ptr<T> ptr;

iterator()

{

index = 0;

}

iterator(const iterator &it)

{

this->vect = it.vect;

this->ptr = it.ptr;

this->index = it.index;

}

iterator(int ind, vector<T> \*vect)

{

this->ptr = std::make\_shared<T>((vect->data[ind]));

this->index = ind;

this->vect = vect;

}

iterator operator++()

{

//cout<<this<<" "<<this->index<<" "<<this->vect->size<<endl;

if ((this->index + 1) > this->vect->size)

{

//printf("Iterator cannot be incremented past the end of range.%d %d\n", this->index, this->vect->size);

throw std::out\_of\_range("Iterator cannot be incremented past the end of range.");

return \*this;

}

this->index++;

this->ptr = std::make\_shared<T>(this->vect->data[this->index]);

return \*this;

}

iterator &operator++(int)

{

if ((this->index + 1) > (this->vect->size))

{

//printf("Iterator cannot be incremented past the end of range.\n");

throw std::out\_of\_range("Iterator cannot be incremented past the end of range.");

return \*this;

}

this->index++;

this->ptr = std::make\_shared<T>(this->vect->data[this->index]);

return \*this;

}

iterator operator=(iterator x)

{

//printf("=\n");

this->ptr = x.ptr;

return \*this;

}

bool operator!=(const iterator &x)

{ //printf("!=\n");

if (index != x.index)

return true;

//printf("false\n ");

/\* if(ptr != x.ptr) return true;

printf("\t !=\n "); \*/

return false;

}

bool operator>=(iterator x)

{

return this->ptr >= x.ptr;

}

int operator-(iterator x)

{

//printf("-\n");

return this->index - x.index;

}

bool operator==(const iterator &x)

{

//printf("\t ==\n ");

if (index == x.index)

return true;

if (ptr == x.ptr)

return true;

return false;

}

T operator\*()

{

return \*(this->ptr);

}

T \*operator->()

{

return this->ptr;

}

~iterator() {}

};

iterator begin()

{

vector<T>::iterator it(0, this);

return it;

}

iterator end()

{

vector<T>::iterator it(this->size, this);

return it;

}

void Printing()

{

printf(">>>>>>>>\n");

for (int i = 0; i < this->size; i++)

{

cout << this->data[i] << " ";

}

cout << endl;

printf(">>>>>>>>size %d | cap %d\n", this->size, this->capacity);

}

void Show()

{

printf(">>>>>>>>\n");

for (int i = 0; i < this->capacity; i++)

{

cout << this->data[i] << " ";

}

cout << endl;

printf(">>>>>>>>size %d | cap %d\n", this->size, this->capacity);

}

~vector()

{

this->size = 0;

this->capacity = 0;

delete[] this->data;

}

T operator[](int i)

{

return this->data[i];

}

T \*operator++(int x)

{

return this->data[x + 1];

}

T \*operator--(int x)

{

return this->data[x - 1];

}

void operator=(vector<T> vec)

{

for (int i = 0; i < this->size; i++)

{

this->data[i] = vec.data[i];

}

}

};

template <class T, size\_t BLOCK\_SIZE>

class Alloc

{ //класс аллокатора

public:

T \*allocate(int n = 1) //конструктор аллокатора и выделение памяти

{

int idx;

T \*arr;

if (last < BLOCK\_SIZE)

{

idx = last;

arr = &block[idx];

last += n;

}

else

{

if (!free.Empty())

{

if (this->PlaceOK(n))

{

arr = &block[idx];

for (int i = 0; i < n; i++)

{

idx = free.PopEnd();

}

}

else

{

this->MoveData();

}

}

else

{

throw std::runtime\_error("Allocator has no more free blocks");

}

}

return arr;

}

void deallocate(T \*ptr, int n = 1) //деструктор аллокатора

{

for (int i = 0; i < n; i++)

{

int idx = ptr + i \* sizeof(T) - &block[0];

free.PushEnd(idx);

}

}

private:

T block[BLOCK\_SIZE]; //размер аллокатора

vector<int> free; //Освобождение памяти из стека

int last = 0; //наименьший из нераспределенных индексов

bool PlaceOK(int n)

{

int ind = free.size - 1;

for (int i = 0; i < n; i++)

{

if (free[ind] - 1 != free[ind - 1])

{

return false;

}

}

return true;

}

void MoveData()

{

printf("Двигаем память аллокатором для удобного распределения\n");

}

};

/\* std::shared\_ptr<Node> head;//головной узел

std::weak\_ptr<Node> tail;//хвостовой узел

size\_t size;//размер

static const size\_t memSize = 10000;//фиксированный размер аллокатора

Alloc<Node,memSize> alloc;//конструктор аллокатора

void deallocNode( Node \*ptr ) {//деструктор аллокатора

alloc.Alloc::deallocate( ptr );

}

std::shared\_ptr<Node> allocNode(const T& val) {//Выделение памяти из аллокатора на узел очереди

Node \*res = alloc.Alloc::allocate();

res->value = val;//

return std::shared\_ptr<Node>( res, std::bind( &Queue::deallocNode, this, std::placeholders::\_1 ));

}

\*/

**For file triangle.hpp:**

template <typename T>

struct Triangle //: public Figure<T>

{

private:

bool MakeTr(TP x, TP y, TP x1, TP y1)

{

double angle = 3.14 / 180.0 \* (360 / 3);

for (int i = 0; i < 3; i++)

{

this->a[i].first = x;

this->a[i].second = y;

T nextX = ((x1 - x) \* cos(angle) + (y1 - y) \* -sin(angle)) + x1;

T nextY = ((x1 - x) \* sin(angle) + (y1 - y) \* cos(angle)) + y1;

x = x1;

y = y1;

x1 = nextX;

y1 = nextY;

}

}

public:

int count = 3;

std::pair<T, T> a[3];

Triangle()

{

this->MakeTr(0, 0, 1, 1);

}

Triangle(T x, T y, T x1, T y1)

{

this->MakeTr(x, y, x1, y1);

}

Triangle(Triangle<T> &tr)

{

this->a[0] = tr.a[0];

this->a[1] = tr.a[1];

this->a[2] = tr.a[2];

}

void Print()

{

for (int i = 0; i < 3; i++)

{

cout << "Вершина " << i + 1 << " x = " << this->a[i].first << " y = " << this->a[i].second << endl;

}

}

double Square()

{

double s, a, h;

a = sqrt((double)((this->a[0].first - this->a[1].first) \* (this->a[0].first - this->a[1].first) + (this->a[0].second - this->a[1].second) \* (this->a[0].second - this->a[1].second)));

//a = sqrt((x1-x2)^2+(y1-y2)^2)

h = a \* sin(3.14 / 3);

s = (a \* h) / 2;

return s;

}

void operator=(Triangle<TP> tr)

{

for (int i = 0; i < 3; i++)

{

this->a[i].first = tr.a[i].first;

this->a[i].second = tr.a[i].second;

}

}

~Triangle() {}

};

**Вывод**

Была реализована программа, в которой применяются основы работы с коллекциями и основы работы с аллокаторами памяти. Научились пользоваться аллокаторами. Закрепили навыки работы с реализовывать шаблонами классов и подключением файлов к программе.

**Литература:**

1. Классы с++ [Электронный ресурс]URL: <https://codelessons.ru/cplusplus/klassy-v-c-rukovodstvo-dlya-nachinayushhix.html> (Дата обращения 05.12.20).
2. Умные указатели для начинающих [Электронный ресурс] URL:  
   <https://habr.com/ru/post/140222/> (Дата обращения 05.12.20).
3. STL Итераторы С++ URL[Электронный ресурс]: <https://ravesli.com/urok-198-iteratory-stl/#toc-2> (Дата обращения 18.11.20).
4. Шаблоны и шаблонные функции в C++[Электронный ресурс]:<https://code-live.ru/post/cpp-template-functions/> (Дата обращения 05.12.20).
5. Аллокаторы памяти - Habr [Электронный ресурс] URL: https://habr.com/ru/post/505632/ (Дата обращения 05.12.20).