

**Московский авиационный институт
(национальный исследовательский университет)**

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №4 по курсу «ООП»

Студент: Носов Э. Л.
Преподаватель:
Группа: М8О-308Б-21
Дата:
Оценка:
Подпись:

Москва, 2024

Лабораторная работа №4

Цель:

- Закрепление навыков работы с классами;
- Создание простых динамических структур данных;
- Работа с объектами, передаваемыми «по значению»

Вариант №29: Необходимо спроектировать и запрограммировать на языке C++ класс-контейнер очередь (TQueue), содержащий одну фигуру (класс Triangle). Классы должны удовлетворять следующим правилам:

- Требования к классу фигуры аналогичны требованиям из лабораторной работы №3;
- Классы фигур должны содержать набор следующих методов:
 - Перегруженный оператор ввода координат вершин фигуры из потока `std::istream (>)`; Он должен заменить конструктор, принимающий координаты вершин из стандартного потока;
 - Перегруженный оператор вывода в поток `std::ostream (<)`, заменяющий метод `Print` из лабораторной работы 1;
 - Оператор копирования `(=)`;
 - Оператор сравнения с такими же фигурами `(==)`.
- Класс-контейнер должен содержать объекты фигур ‘по значению’ (не по ссылке);
- Класс-контейнер должен содержать набор следующих методов:
 - Метод по добавлению фигуры в контейнер (`Push`);
 - Метод по получению фигуры из контейнера (`Top`);
 - Метод по удалению фигуры из контейнера (`Pop`).

Программа должна позволять:

- Вводить произвольное количество фигур и добавлять их в контейнер;
- Распечатывать содержимое контейнера;
- Удалять фигуры из контейнера

Ссылка на репозиторий: https://github.com/nsveml/UniProjects/tree/main/OOP/oop_exercise_4

1 Исходный код

Классы фигур реализованы так же как в предыдущей работе.

Класс контейнер TQueue определен в заголовочном файле queue.h, в нем указаны приватные атрибуты: целочисленные size и capacity, указатель на класс фигуры storage, в нем же объявлены методы класса, а именно публичные Top, Pop, Push, Empty, Length, Clear, конструкторы и деструктор, приватный метод Extend, необходимый для реаллокации памяти, и перегрузка оператора вывода. Определение всех этих функций вынесено в файл queue.cpp.

В файле main.cpp реализовано взаимодействие с пользователем, позволяющее создавать фигуры заданного класса и помещать их в контейнер TQueue.

figure.h

```
1 | #include<iostream>
2 |
3 | #ifndef figure_h
4 | #define figure_h
5 |
6 | class Figure{
7 |     public:
8 |
9 |     Figure();
10 |    Figure(int a);
11 |    virtual ~Figure();
12 |    virtual size_t VertexesNumber();
13 |    virtual float Area();
14 |    virtual void Print(std::ostream& os);
15 |
16 |    protected:
17 |
18 |    size_t n;
19 |    float* coords;
20 |    const char* name = "Figure";
21 |
22 | };
23 |
24 | #include"figure.cpp"
25 |
26 | #endif
```

figure.cpp

```
1 | Figure::Figure(){
2 |     n=0;
3 |     coords=nullptr;
4 | }
5 |
6 | Figure::Figure(int a){
7 |     n=a;
8 |     coords = new float [2*n];
9 |     std::cout<<"Enter coordinates in clockwise order:\n";
10 |    for(long unsigned int i = 0; i<2*n; i++){
11 |        std::cin>>coords[i];
12 |    }
13 | }
14 |
15 | Figure::~Figure(){
```

```

16 |     delete [] coords;
17 | }
18 |
19 | size_t Figure::VertexesNumber(){
20 |     return n;
21 | }
22 |
23 | float Figure::Area(){
24 |     float area = 0;
25 |     for(long unsigned int i=0; i<2*n; i++){
26 |         area += (i % 2) ? -coords[i]*coords[(i+1) % (2*n)] : coords[i]*coords[(i+3) % (2*n)];
27 |     }
28 |     area/=2;
29 |     return (area<0)? -area : area;
30 | }
31 |
32 | void Figure::Print(std::ostream& os){
33 |     os<<name<<": ";
34 |     for(long unsigned int i = 0; i<2*n; i+=2){
35 |         os<<" ("<<coords[i]<<", "<<coords[i+1]<<')';
36 |     }
37 |     os<<'\n';
38 | }

```

triangle.h

```

1 | #include"figure.h"
2 |
3 | class Triangle : public Figure{
4 |     public:
5 |
6 |     Triangle();
7 |
8 |     Triangle(std::istream& is);
9 |
10 | };
11 |
12 | #include"triangle.cpp"

```

triangle.cpp

```

1 | Triangle::Triangle(){
2 |     name = "Triangle";
3 |     n=3;
4 |     coords = new float [2*n];
5 |     for(long unsigned int i = 0; i<2*n; i++){
6 |         coords[i]=0.0;
7 |     }
8 | }
9 |
10 | Triangle::Triangle(std::istream& is){
11 |     name = "Triangle";
12 |     n=3;
13 |     coords = new float[2*n];
14 |     for(long unsigned int i=0; i<2*n; i++){
15 |         is>>coords[i];
16 |     }
17 | }

```

queue.h

```

1 | #ifndef QUEUE_H
2 | #define QUEUE_H

```

```

3
4 class TQueue {
5     public:
6         TQueue();
7         TQueue(const TQueue& other);
8         void Push(const Triangle& polygon);
9         const Triangle Pop();
10        const Triangle& Top();
11        bool Empty();
12        size_t Length();
13        friend std::ostream& operator<<(std::ostream& os, const TQueue& queue);
14        void Clear();
15        virtual ~TQueue();
16
17    protected:
18        int size, capacity;
19        Triangle* storage;
20        void extend();
21
22 };
23
24 #include"queue.cpp"
25
26 #endif

```

queue.cpp

```

1
2 TQueue::TQueue(){
3     size = 0;
4     capacity = 1;
5     storage = new Triangle[1];
6 }
7
8 TQueue::TQueue(const TQueue& other){
9     size = other.size;
10    capacity = other.capacity;
11    storage = new Triangle[capacity];
12    for(int i = 0; i < size; i++){
13        storage[i]=other.storage[i];
14    }
15 }
16
17 void TQueue::Push(const Triangle& polygon){
18     if(capacity == size) extend();
19     storage[size]=polygon;
20     size++;
21 }
22
23 const Triangle TQueue::Pop(){
24     if(size > 0){
25         size--;
26         return(storage[size]);
27     }
28     else{
29         return Triangle();
30     }
31 }
32
33 const Triangle& TQueue::Top(){
34     return storage[size-1];
35 }
36

```

```

37 | bool TQueue::Empty(){
38 |     return bool(size);
39 | }
40 |
41 | size_t TQueue::Length(){
42 |     return size;
43 | }
44 |
45 | std::ostream& operator<<(std::ostream& os, const TQueue& queue){
46 |     for(int i = 0; i < queue.size; i++){
47 |         os << queue.storage[i].Area() << ' ';
48 |     }
49 |     os << "\n";
50 |     return os;
51 | }
52 |
53 | void TQueue::Clear(){
54 |     size = 0;
55 |     capacity = 1;
56 |     delete [] storage;
57 |     storage = new Triangle[1];
58 | }
59 |
60 | TQueue::~TQueue(){
61 |     delete [] storage;
62 | }
63 |
64 | void TQueue::extend(){
65 |     Triangle* tmp;
66 |     capacity*=2;
67 |     tmp = new Triangle[capacity];
68 |     for(int i = 0; i < size; i++){
69 |         tmp[i]=storage[i];
70 |     }
71 |     delete [] storage;
72 |     storage = tmp;
73 | }

```

main.cpp

```

1 | #include<iostream>
2 | #include"triangle.h"
3 | #include"queue.h"
4 |
5 | int main(){
6 |     std::cout<<"\
7 |     u - push\n\
8 |     o - pop\n\
9 |     t - top\n\
10 |    i - print\n\
11 |    c - clear\n\
12 |    q - quit\n";
13 |
14 |     Triangle triangle;
15 |     TQueue triangles;
16 |
17 |     std::cout << Triangle();
18 |     char a=' ';
19 |
20 |     while(a!='q'){
21 |         std::cout << "Enter commands:\n";
22 |         std::cin>>a;
23 |         switch(a){

```

```

24         case 'u':
25             std::cout << "Enter coordinates in clockwise order:\n";
26             std::cin >> triangle;
27             triangles.Push(triangle);
28             break;
29         case 'o':
30             std::cout << triangles.Pop();
31             break;
32         case 't':
33             std::cout << triangles.Top();
34             break;
35         case 'i':
36             std::cout << triangles;
37             break;
38         case 'c':
39             triangles.Clear();
40             std::cout << triangles;
41         case 'q':
42             break;
43     }
44 }
45 }

```

2 Тестовые данные

```
nsveml@LAPTOP-LNCHGOM3:/mnt/d/X-Files/MAI/3 sem/OOP/labs/oop_exercise_4$ cat test1
```

```
u
1 1
0 0
1 0
u
2 2
0 0
1 0
u
3 3
0 0
1 0
u
4 4
0 0
1 0
i
```

```
qnsveml@LAPTOP-LNCHGOM3:/mnt/d/X-Files/MAI/3 sem/OOP/labs/oop_exercise_4$ cat test2
```

```
u
1 1
0 0
1 0
t
u
2 2
0 0
1 0
t
u
3 3
0 0
1 0
t
i
u
4 4
0 0
1 0
t
i
o
i
```

```
qnsveml@LAPTOP-LNCHGOM3:/mnt/d/X-Files/MAI/3 sem/OOP/labs/oop_exercise_4$ cat test3
```

```
u
1 1
0 0
```


1 0
t
u
2 2
0 0
1 0
t
u
3 3
0 0
1 0
t
u
4 4
0 0
1 0
t
i
c
i

3 Работа программы

```
nsveml@LAPTOP-LNCHGOM3:/mnt/d/X-Files/MAI/3 sem/OOP/labs/oop_exercise_4$ mkdir cm
nsveml@LAPTOP-LNCHGOM3:/mnt/d/X-Files/MAI/3 sem/OOP/labs/oop_exercise_4$ cd cm
nsveml@LAPTOP-LNCHGOM3:/mnt/d/X-Files/MAI/3 sem/OOP/labs/oop_exercise_4/cm$ cmake ../
CMake Deprecation Warning at CMakeLists.txt:1 (cmake_minimum_required):
  Compatibility with CMake < 2.8.12 will be removed from a future version of
  CMake.
```

Update the VERSION argument <min> value or use a ...<max> suffix to tell CMake that the project does not need compatibility with older versions.

```
-- The C compiler identification is GNU 12.0.1
-- The CXX compiler identification is GNU 12.0.1
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: /usr/bin/cc - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: /usr/bin/c++ - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: /mnt/d/X-Files/MAI/3 sem/OOP/labs/oop_exercise_4/cm
nsveml@LAPTOP-LNCHGOM3:/mnt/d/X-Files/MAI/3 sem/OOP/labs/oop_exercise_4/cm$ make
[ 50%] Building CXX object CMakeFiles/oop_exercise_4.dir/main.cpp.o
[100%] Linking CXX executable "/mnt/d/X-Files/MAI/3 sem/OOP/labs/oop_exercise_4/oop_exercise_4"
[100%] Built target oop_exercise_4
nsveml@LAPTOP-LNCHGOM3:/mnt/d/X-Files/MAI/3 sem/OOP/labs/oop_exercise_4/cm$ cd ../; ls
CMakeLists.txt  cm  headers  main.cpp  oop_exercise_4  test1  test2  test3
nsveml@LAPTOP-LNCHGOM3:/mnt/d/X-Files/MAI/3 sem/OOP/labs/oop_exercise_4$ ./oop_exercise_4 < test1
  u - push
  o - pop
  t - top
  i - print
  c - clear
  q - quit
Triangle: (0, 0) (0, 0) (0, 0)
Enter commands:
Enter coordinates in clockwise order:
Enter commands:
Enter coordinates in clockwise order:
Enter commands:
Enter coordinates in clockwise order:
Enter commands:
```

```

Enter coordinates in clockwise order:
Enter commands:
0.5 1 1.5 2 =>
Enter commands:
nsveml@LAPTOP-LNCHGOM3:/mnt/d/X-Files/MAI/3 sem/OOP/labs/oop_exercise_4$ ./oop_exercise_4 < test2
    u - push
    o - pop
    t - top
    i - print
    c - clear
    q - quit
Triangle: (0, 0) (0, 0) (0, 0)
Enter commands:
Enter coordinates in clockwise order:
Enter commands:
Triangle: (1, 1) (0, 0) (1, 0)
Enter commands:
Enter coordinates in clockwise order:
Enter commands:
Triangle: (2, 2) (0, 0) (1, 0)
Enter commands:
Enter coordinates in clockwise order:
Enter commands:
Triangle: (3, 3) (0, 0) (1, 0)
Enter commands:
0.5 1 1.5 =>
Enter commands:
Enter coordinates in clockwise order:
Enter commands:
Triangle: (4, 4) (0, 0) (1, 0)
Enter commands:
0.5 1 1.5 2 =>
Enter commands:
Figure: (4, 4) (0, 0) (1, 0)
Enter commands:
0.5 1 1.5 =>
Enter commands:
nsveml@LAPTOP-LNCHGOM3:/mnt/d/X-Files/MAI/3 sem/OOP/labs/oop_exercise_4$ ./oop_exercise_4 < test3
    u - push
    o - pop
    t - top
    i - print
    c - clear
    q - quit
Triangle: (0, 0) (0, 0) (0, 0)
Enter commands:
Enter coordinates in clockwise order:
Enter commands:
Triangle: (1, 1) (0, 0) (1, 0)

```

Enter commands:
Enter coordinates in clockwise order:
Enter commands:
Triangle: (2, 2) (0, 0) (1, 0)
Enter commands:
Enter coordinates in clockwise order:
Enter commands:
Triangle: (3, 3) (0, 0) (1, 0)
Enter commands:
Enter coordinates in clockwise order:
Enter commands:
Triangle: (4, 4) (0, 0) (1, 0)
Enter commands:
0.5 1 1.5 2 =>
Enter commands:
=>
Enter commands:
=>
Enter commands:

4 Вывод

Выполнив четвертую лабораторную работу по курсу «ООП», я закрепил полученные ранее знания и реализовал динамическую структуру данных, хранящую объекты "по значению".