**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 3**

Тема: Механизмы наследования в С++

Студент: Гребенков Дмитрий Игоревич

Группа: 80-207

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2020

1. Постановка задачи

Разработать классы согласно варианту задания, классы должны наследоваться от базового класса Figure.

Все классы должны поддерживать набор общих методов:

1. Вычисление геометрического центра фигуры.

2. Вывод в стандартный поток вывода std::cout координат вершин фигуры;

3. Вычисление площади фигуры.

Создать программу, которая позволяет:

• Вводить из стандартного ввода std::cin фигуры, согласно варианту задания.

• Сохранять созданные фигуры в динамический массив std::vector<Figure\*>.

• Вызывать для всего массива общие функции (1-3 см. выше).Т.е. распечатывать для каждой фигуры в массиве геометрический центр, координаты вершин и площадь.

• Необходимо уметь вычислять общую площадь фигур в массиве.

• Удалять из массива фигуру по индексу.

Вариант задания: 12. Фигуры: трапеция, ромб, 5-угольник.

1. Описание программы

Создан класс Figure с виртуальными методами вычисления геометрического центра фигуры, вывода координат вершин фигуры и вычисления площади фигуры. Добавлены классы, дочерние для Figure и содержащие необходимые для вычислений методы и переменные для хранения фигур.

Реализована возможность добавления фигуры с новыми параметрами в вектор. В этом векторе можно хранить различные фигуры. Также присутствует возможность вычисления общей площади фигур в массиве, вывода данных обо всех фигурах и удаления по индексу.

Взаимодействие с пользователем осуществляется при помощи меню.

Примечание: для ускорения вычислений и упрощения хранения на вводимые фигуры установлены некоторые ограничения. Во-первых, основания вводимых трапеций должны быть параллельны оси Ox. Во-вторых, большее основание трапеции должно быть снизу. В-третьих, диагонали вводимых ромбов должны быть параллельны осям Ox и Oy.

Трапеция задается координатами трех вершин.

Ромб задается координатами двух вершин.

Пятиугольник задается центром описанной окружности и радиусом.

1. Набор тестов и результаты выполнения

Menu:

0. Print menu.

1. Enter trapezoid.

2. Enter rhombus.

3. Enter pentagon.

4. Print center, vertices and square of all figures.

5. Print sum of squares of all figures in array.

6. Delete figure.

7. Quit the program.

Choose option: 1

Enter coordinates of left-top, left-bottom and right-bottom vertices:

1 3 0 0 5 0

Choose option: 4

Figure [0]:

Center: (2.5, 1.375);

Vertices of this figure:

(1, 3)

(0, 0)

(5, 0)

(4, 3)

Square: 12

Choose option: 2

Enter coordinates of left and bottom vertices: 0 1 1 0

Choose option: 4

Figure [0]:

Center: (2.5, 1.375);

Vertices of this figure:

(1, 3)

(0, 0)

(5, 0)

(4, 3)

Square: 12

Figure [1]:

Center: (1, 1);

Vertices of this figure:

(0, 1)

(1, 0)

(2, 1)

(1, 2)

Square: 2

Choose option: 5

Sum of squares of all figures in array: 14

Choose option: 2

Enter coordinates of left and bottom vertices: -1 0 0 -2

Choose option: 1

Enter coordinates of left-top, left-bottom and right-bottom vertices:

3 4 1 1 8 1

Choose option: 4

Figure [0]:

Center: (2.5, 1.375);

Vertices of this figure:

(1, 3)

(0, 0)

(5, 0)

(4, 3)

Square: 12

Figure [1]:

Center: (1, 1);

Vertices of this figure:

(0, 1)

(1, 0)

(2, 1)

(1, 2)

Square: 2

Figure [2]:

Center: (0, 0);

Vertices of this figure:

(-1, 0)

(0, -2)

(1, 0)

(0, 2)

Square: 4

Figure [3]:

Center: (4.5, 2.3);

Vertices of this figure:

(3, 4)

(1, 1)

(8, 1)

(6, 4)

Square: 15

Choose option: 5

Sum of squares of all figures in array: 33

Choose option: 3

Enter coordinates of center: 5 4

Enter radius of the circumscribed circle: 2

Choose option: 4

Figure [0]:

Center: (2.5, 1.375);

Vertices of this figure:

(1, 3)

(0, 0)

(5, 0)

(4, 3)

Square: 12

Figure [1]:

Center: (1, 1);

Vertices of this figure:

(0, 1)

(1, 0)

(2, 1)

(1, 2)

Square: 2

Figure [2]:

Center: (0, 0);

Vertices of this figure:

(-1, 0)

(0, -2)

(1, 0)

(0, 2)

Square: 4

Figure [3]:

Center: (4.5, 2.3);

Vertices of this figure:

(3, 4)

(1, 1)

(8, 1)

(6, 4)

Square: 15

Figure [4]:

Center: (5, 4);

Vertices of this figure:

(3.09789, 4.61804)

(5, 6)

(6.90211, 4.61804)

(6.17557, 2.38197)

(3.82443, 2.38197)

Square: 9.51056

Choose option: 5

Sum of squares of all figures in array: 42.5106

Choose option: 6

Enter index: 0

Choose option: 6

Enter index: 1

Choose option: 4

Figure [0]:

Center: (1, 1);

Vertices of this figure:

(0, 1)

(1, 0)

(2, 1)

(1, 2)

Square: 2

Figure [1]:

Center: (4.5, 2.3);

Vertices of this figure:

(3, 4)

(1, 1)

(8, 1)

(6, 4)

Square: 15

Figure [2]:

Center: (5, 4);

Vertices of this figure:

(3.09789, 4.61804)

(5, 6)

(6.90211, 4.61804)

(6.17557, 2.38197)

(3.82443, 2.38197)

Square: 9.51056

Choose option: 6

Enter index: 3

Error: incorrect index.

Choose option: 6

Enter index: 2

Choose option: 6

Enter index: 1

Choose option: 6

Enter index: 0

Choose option: 4

Array is empty.

Choose option: 6

Enter index: 0

Error: incorrect index.

Choose option: 5

Error: array is empty.

Choose option: 0

Menu:

0. Print menu.

1. Enter trapezoid.

2. Enter rhombus.

3. Enter pentagon.

4. Print center, vertices and square of all figures.

5. Print sum of squares of all figures in array.

6. Delete figure.

7. Quit the program.

Choose option: 7

1. Листинг программы

/\*

Гребенков Д. И.

Группа М8О-207Б-19

Вариант 12

Разработать классы согласно варианту задания, классы должны наследоваться от базового класса Figure.

Все классы должны поддерживать набор общих методов:

1. Вычисление геометрического центра фигуры;

2. Вывод в стандартный поток вывода std::cout координат вершин фигуры;

3. Вычисление площади фигуры.

Создать программу, которая позволяет:

• Вводить из стандартного ввода std::cin фигуры, согласно варианту задания.

• Сохранять созданные фигуры в динамический массив std::vector<Figure\*>.

• Вызывать для всего массива общие функции (1-3 см. выше).Т.е. распечатывать для каждой фигуры в массиве

геометрический центр, координаты вершин и площадь.

• Необходимо уметь вычислять общую площадь фигур в массиве.

• Удалять из массива фигуру по индексу.

Фигуры: трапеция, ромб, 5-угольник.

\*/

#include <iostream>

#include <vector>

#include <cmath>

const double pi = 3.14159;

class Coordinates {

public:

double x;

double y;

Coordinates () {}

Coordinates (double a, double b) {

x = a;

y = b;

}

};

class Figure {

public:

virtual Coordinates centre () = 0;

virtual void vertices () = 0;

virtual double square () = 0;

virtual ~Figure () {};

};

class Trapezoid: public Figure {

private:

Coordinates lt;

Coordinates lb;

Coordinates rb;

public:

Trapezoid (Coordinates a, Coordinates b, Coordinates c) {

lt = a;

lb = b;

rb = c;

}

virtual Coordinates centre () {

Coordinates result;

result.x = lb.x + ((rb.x - lb.x) / 2.0);

result.y = lt.y - (((lt.y - lb.y) \* (3 \* rb.x - lb.x - 2 \* lt.x)) /

(6.0 \* (rb.x - lt.x)));

return result;

}

virtual void vertices () {

std::cout << "(" << lt.x << ", " << lt.y << ")" << std::endl;

std::cout << "(" << lb.x << ", " << lb.y << ")" << std::endl;

std::cout << "(" << rb.x << ", " << rb.y << ")" << std::endl;

Coordinates rt (rb.x - lt.x + lb.x, lt.y);

std::cout << "(" << rt.x << ", " << rt.y << ")" << std::endl;

}

virtual double square () {

return ((rb.x - lt.x) \* (lt.y - lb.y));

}

virtual ~Trapezoid () {};

};

class Rhombus: public Figure {

private:

Coordinates l;

Coordinates b;

public:

Rhombus (Coordinates v1, Coordinates v2) {

l = v1;

b = v2;

}

virtual Coordinates centre () {

return (Coordinates (b.x, l.y));

}

virtual void vertices () {

std::cout << "(" << l.x << ", " << l.y << ")" << std::endl;

std::cout << "(" << b.x << ", " << b.y << ")" << std::endl;

std::cout << "(" << 2 \* b.x - l.x << ", " << l.y << ")" << std::endl;

std::cout << "(" << b.x << ", " << 2 \* l.y - b.y << ")" << std::endl;

}

virtual double square () {

return (2 \* (b.x - l.x) \* (l.y - b.y));

}

virtual ~Rhombus () {};

};

class Pentagon: public Figure {

private:

Coordinates center;

double radius;

public:

Pentagon (Coordinates c, double r) {

center = c;

radius = r;

}

virtual Coordinates centre () {

return center;

}

virtual void vertices () {

std::cout << "(" << center.x - radius \* sin (72 \* pi / 180) << ", " <<

center.y + radius \* cos (72 \* pi / 180) << ")" << std::endl;

std::cout << "(" << center.x << ", " << center.y + radius << ")" << std::endl;

std::cout << "(" << center.x + radius \* sin (72 \* pi / 180) << ", " <<

center.y + radius \* cos (72 \* pi / 180) << ")" << std::endl;

std::cout << "(" << center.x + radius \* sin (36 \* pi / 180) << ", " <<

center.y - radius \* cos (36 \* pi / 180) << ")" << std::endl;

std::cout << "(" << center.x - radius \* sin (36 \* pi / 180) << ", " <<

center.y - radius \* cos (36 \* pi / 180) << ")" << std::endl;

}

virtual double square () {

return (5 \* pow (radius, 2) \* sin (72 \* pi / 180.0) / 2.0);

}

virtual ~Pentagon () {};

};

void show\_options () {

std::cout << "Menu:" << std::endl;

std::cout << "0. Print menu." << std::endl;

std::cout << "1. Enter trapezoid." << std::endl;

std::cout << "2. Enter rhombus." << std::endl;

std::cout << "3. Enter pentagon." << std::endl;

std::cout << "4. Print center, vertices and square of all figures." << std::endl;

std::cout << "5. Print sum of squares of all figures in array." << std::endl;

std::cout << "6. Delete figure." << std::endl;

std::cout << "7. Quit the program." << std::endl;

}

int main () {

std::vector <Figure\*> data;

int option = 0;

show\_options ();

while (option != 7) {

std::cout << "Choose option: ";

std::cin >> option;

switch (option) {

case 0: {

show\_options ();

break;

}

case 1: {

Coordinates a, b, c;

std::cout << "Enter coordinates of left-top, left-bottom and right-bottom vertices: "

<< std::endl;

std::cin >> a.x >> a.y >> b.x >> b.y >> c.x >> c.y;

Trapezoid\* alpha = new Trapezoid (a, b, c);

data.push\_back (alpha);

break;

}

case 2: {

Coordinates a, b;

std::cout << "Enter coordinates of left and bottom vertices: ";

std::cin >> a.x >> a.y >> b.x >> b.y;

Rhombus\* beta = new Rhombus (a, b);

data.push\_back (beta);

break;

}

case 3: {

Coordinates o;

double r;

std::cout << "Enter coordinates of center: ";

std::cin >> o.x >> o.y;

std::cout << "Enter radius of the circumscribed circle: ";

std::cin >> r;

Pentagon\* gamma = new Pentagon (o, r);

data.push\_back (gamma);

break;

}

case 4: {

if (data.empty () != true) {

for (unsigned int i = 0; i < data.size (); i ++) {

std::cout << "Figure [" << i << "]:" << std::endl;

std::cout << "Center: (" << data [i]->centre ().x <<

", " << data [i]->centre ().y << ");" << std::endl;

std::cout << "Vertices of this figure:" << std::endl;

data [i]->vertices ();

std::cout << "Square: " << data [i]->square () << std::endl;

}

}

else {

std::cout << "Array is empty." << std::endl;

}

break;

}

case 5: {

if (data.empty () != true) {

double sq = 0;

for (unsigned int i = 0; i < data.size (); i ++) {

sq += data [i]->square ();

}

std::cout << "Sum of squares of all figures in array: " << sq << std::endl;

}

else {

std::cout << "Error: array is empty." << std::endl;

}

break;

}

case 6: {

int idx;

std::cout << "Enter index: ";

std::cin >> idx;

if ((idx >= 0) && (idx < int (data.size ()))) {

if (data [idx] != nullptr) {

delete data [idx];

data.erase (data.begin () + idx);

}

}

else {

std::cout << "Error: incorrect index." << std::endl;

}

}

case 7: {

break;

}

default: {

std::cout << "Error: incorrect option." << std::endl;

}

}

}

for (unsigned int i = 0; i < data.size (); i ++) {

delete data [i];

}

data.clear ();

return 0;

}

1. Вывод

Изучены базовые принципы наследования в C++.

Литература

1. C++ | Наследование [Электронный ресурс] URL: <https://metanit.com/cpp/tutorial/5.10.php> (дата обращения 19.10.2020).
2. Базовое наследование в C++ | Уроки C++ - Ravesli [Электронный ресурс] URL: <https://ravesli.com/urok-154-bazovoe-nasledovanie-v-c/> (дата обращения 19.10.2020).