|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ *Робототехника и комплексная автоматизация*

КАФЕДРА *Системы автоматизированного проектирования (РК-6)*

**ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

по дисциплине: «Объектно-ориентированное программирование»

|  |  |  |
| --- | --- | --- |
| Студент |  | Камалов Антон Павлович |
| Группа |  | РК6-25Б |
| Тип задания |  | Лабораторная работа №1 |
| Вариант |  | Р16 |

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_ Камалов А. П.**

*подпись, дата фамилия, и.о.*

Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_ Родионов С. В.**

*подпись, дата фамилия, и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Москва, 2023 г.*

# Вариант задания

# Разработать объектно-ориентированную программу для вычисления декартовых координат вершин правильного восьмиугольника с заданной длиной стороны, центр которого смещен на юго-восток от начала координат, а две диагонали параллельны осям координат. Длина стороны и смещение центра восьмиугольника должна передаваться программе аргументами командной строки ее вызова, а полученные координаты его вершин должны отображаться в потоке стандартного вывода. Программная реализация вычислений должна быть основана на разработке класса точки с приватными полями для ее декартовых координат, публичными методами доступа к ним и конструктором инициализации их значений. Кроме того, в этом классе нужно предусмотреть статический метод для конструирования точки по ее полярным координатам.

# Описание программы

**Входные данные**: длина стороны правильного восьмиугольника (*side*) и координаты х и у для смещения относительно начала координат (*deltaX и deltaY)*.

**Выходные данные**: координаты вершин правильного восьмиугольника.

**Класс Point**

*Информационные поля:*

\_x (float), \_y (float) – координаты точки;

*Конструкторы и деструкторы:*

Point (float, float) – инициализации;

Point () – по умолчанию;

*Методы:*

float getX() – публичный метод доступа к приватному полю x;

float getY() – публичный метод доступа к приватному полю y;

static Point polar(float, float) – статический метод конструирования точки по полярным координатам;

Программа принимает на вход длину стороны правильного восьмиугольника и смещение центра правильного восьмиугольника. Затем находятся координаты вершин правильного восьмиугольника. Для этого используется класс Point, который описывает точку на плоскости и позволяет работать с полярными и декартовыми координатами. Координаты вершин находятся по формуле, использующей полярные координаты. По ним (*ρ* и *φ*) мы находим декартовы координаты новой точки на плоскости (x, y) следующим образом:

где *ρ* - радиус вектор, *φ* - угол между осью x и радиус-вектором, указанный в радианах. Затем происходит смещение вершин правильного восьмиугольника на заданное смещение, и программа выводит координаты вершин на экран.

**Список литературы**

1. Лекционный и семинарский материалы
2. Бьёрн Страуструп «Программирование: принципы и практика с использованием С++» 2 издание.
3. Бьёрн Страуструп «Язык программирования С++» второе дополненное издание

# Приложение 1

# Текст программы

|  |  |
| --- | --- |
| 01:  02:  03:  04:  05:  06:  07:  08:  09:  10:  11:  12:  13:  14:  15:  16:  17:  18:  19:  20:  21:  22:  23:  24:  25:  26:  27:  28:  29:  30:  31:  32:  33:  34:  35:  36:  37:  38:  39:  40:  41:  42:  43:  44:  45:  46:  47:  48:  48:  49:  50:  51:  52:  53:  54:  55:  56: | class Point {  private:  float \_x, \_y;  Point (float x, float y): \_x(x),\_y(y) {};  public:  Point(): \_x(0), \_y(0) {};  float getX() {return \_x;}  float getY() {return \_y;}  // Статический метод класса  static Point polar(float, float);  };  // Перевод координат точки из полярной в декартову систему  Point Point::polar(float radius, float f) {  Point p(radius\*cos(f), radius\*sin(f));  return p;  };  int main(int argc, char\*\* argv) {  float side = 5.0;  float angle = 0.0;  float pi = acos(-1.0);  float radius;  float deltaX = 1.0, deltaY = -1.0;  Point vertex[8];  int i = 0;  // Проверка на количество аргументов  if (argc > 3) {  side = atof(argv[1]);  deltaX = atof(argv[2]);  deltaY = atof(argv[3]);  } else{  std::cout << "Invalid count of arguments\n";  return -1;  }  // Проверка на расположение точки  if (deltaX <= 0 || deltaY >= 0){  std::cout << "Invalid coordinates of point\n";  return -2;  }  radius = side\*sqrt(4+2\*sqrt(2))/2;  while (i < 8){  // Создание вершины восьмиугольника  vertex[i] = Point::polar(radius,angle);  angle+=(pi/4.0);  i++;  }  do{  // Вывод координат вершин восьмиугольника  printf("Point %d: %f ; %f\n", i, vertex[i].getX()+deltaX, vertex[i].getY()+deltaY);  i--;  }while(i>0);  return 0;  } |

# Приложение 2

Результаты работы программы

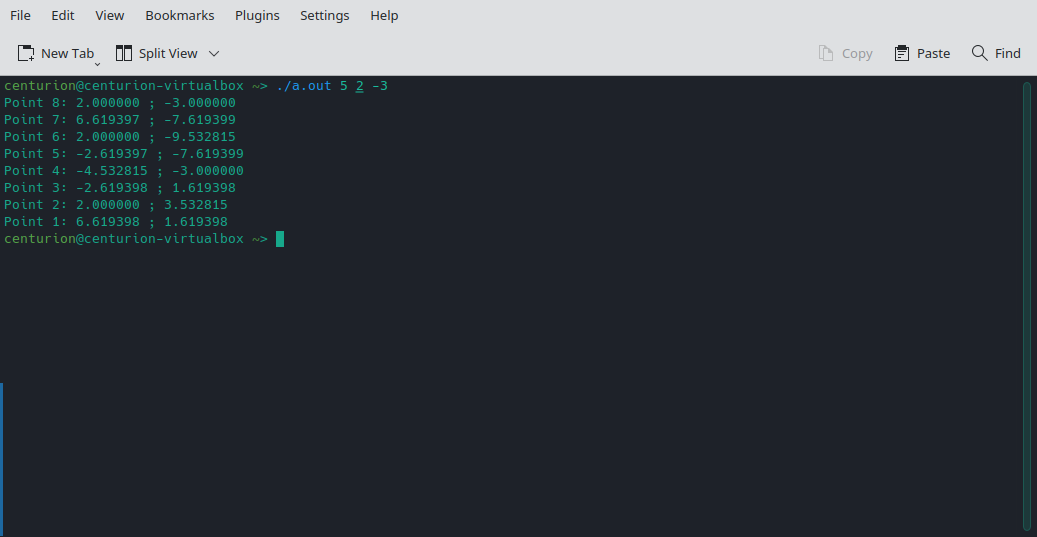


Рисунок 1. Выполнение программы с корректными аргументами

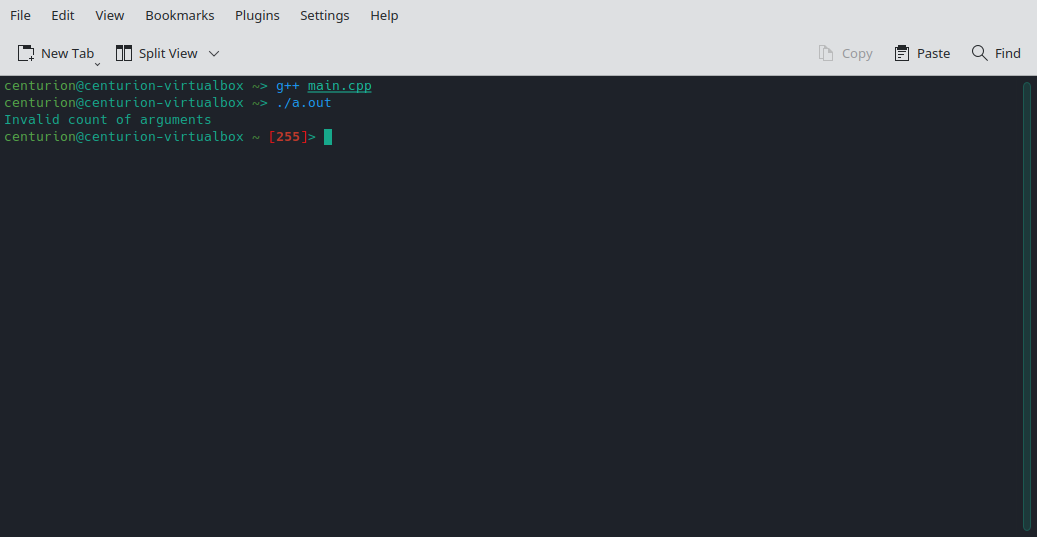


Рисунок 2. Выполнение программы с недостаточным количеством аргументов

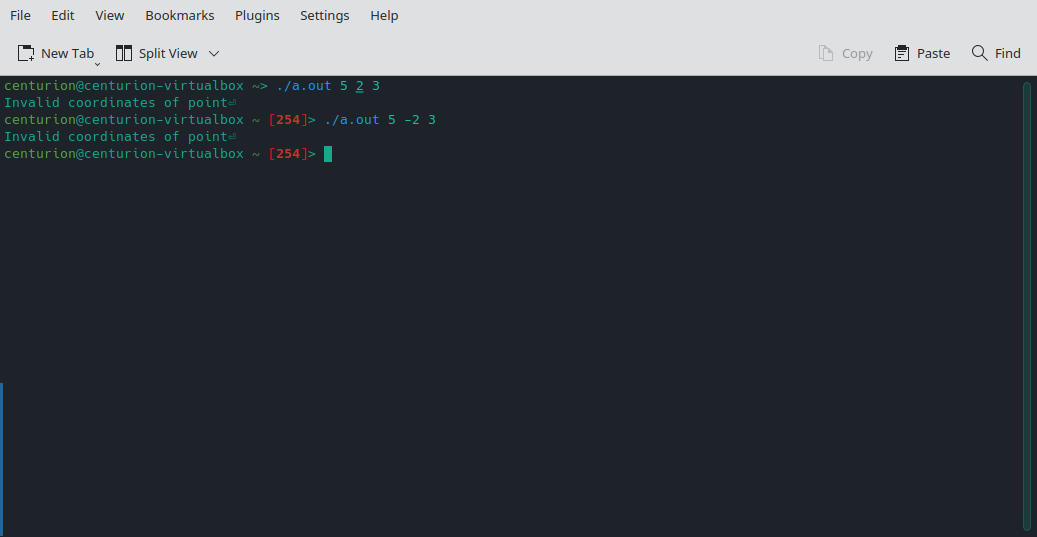


Рисунок 3. Выполнение программы с неправильным расположением центра правильного восьмиугольника