SEG2105 – Assignment 1 Written Answers

Hasan Al-Yabroudi (300256049)

Romuald Bruno Ngansop Kamwoua (300395504)

Introduction

The PointCP family of classes in Java demonstrates various design patterns used for representing points in two-dimensional space, either in Cartesian (x,y) or Polar (rho,theta) coordinates. Four designs were examined, each with its own strengths and weaknesses. These were evaluated for performance in the PointCPPerformanceTest.java file. In the sections below, we provide an analysis of each design along with a discussion on the performance times observed.

Design 1: PointCP

The PointCP class encapsulates both Cartesian and Polar coordinate systems in a single class. It can transform between the two representations using `convertStorageToCartesian()` and `convertStorageToPolar()` methods. This class demonstrates the most versatile approach in terms of functionality.

Advantages
1. Versatility: It can handle both Cartesian and Polar coordinates interchangeably.
2. Flexibility: It provides methods to convert between the two coordinate systems.

Disadvantages
1. Efficiency: It needs to check the current coordinate system every time a getter is called and perform a calculation if necessary, which could impact performance.
2. Complexity: Maintaining two coordinate systems in a single class increases complexity and can lead to potential errors.

Performance (after 10000000 runs): The execution time for PointCP was 1205ms. Its performance is reasonably good, although the need to check the coordinate type every time a getter method is called can lead to additional overhead.

Design 2: PointCP2

PointCP2 class maintains the Polar coordinate representation only. It provides methods to convert to Cartesian coordinates but does not store them.

Advantages
1. Simplicity: It maintains only one set of coordinates, making it less prone to errors.
2. Reduced memory usage: It only requires storage for Polar coordinates.

Disadvantages
1. Performance: Converting to Cartesian coordinates on-demand requires additional computational resources, which may affect performance.
2. Limited functionality: It can only return Cartesian coordinates but cannot store them.

Performance (after 10000000 runs): The execution time for PointCP2 was 2665ms. This is due to the extra computational cost of converting Polar coordinates to Cartesian on-demand.

Design 3: PointCP3

The PointCP3 class only maintains Cartesian coordinates. It provides methods to convert to Polar coordinates but does not store them.

Advantages
1. Simplicity: Like PointCP2, this design also maintains only one set of coordinates, reducing the chance of errors.
2. Reduced memory usage: It only requires storage for Cartesian coordinates.

Disadvantages
1. Performance: Similar to PointCP2, converting to Polar coordinates on-demand requires additional computational resources.
2. Limited functionality: It can only return Polar coordinates but cannot store them.

Performance (after 10000000 runs): The execution time for PointCP3 was 1200ms. This class performs similarly to PointCP in terms of execution speed.

Design 5: PointCP5

The PointCP5 class uses an abstract base class to provide common functionality, and then implements two subclasses: one for Polar coordinates (PointCP2) and another for Cartesian coordinates (PointCP3). This design leverages the principles of inheritance and polymorphism.

Advantages
1. Organized code structure: This design follows the Object-Oriented Programming principles, separating the functionalities into subclasses based on their behavior.
2. Extensibility: New coordinate systems or functions can be added easily by extending the base class.

Disadvantages
1. Increased complexity: The use of inheritance and polymorphism might complicate the design, especially for larger projects.
2. Performance: Creation of additional objects during runtime might increase execution time.

Performance (after 10000000 runs): The execution time for PointCP5 was 3804ms. The higher running time

Sample Outputs:

Design 2

```
hasanalyabroudi@Hasans-MacBook-Pro pointcp % java PointCPTest
Cartesian-Polar Coordinates Conversion Program
Enter the design you want to test (Design2 / Design3 / Design5): Design2

Enter the value of Rho using a decimal point(.): 10
Enter the value of Theta using a decimal point(.): 5

Testing Design 2 (PointCP2):

You entered:
Stored as Polar [10.0,5.0]


After asking to store as Polar:
Stored as Polar [10.0,5.0]

Cartesian coordinates: (9.961946980917455, 0.8715574274765816)

This is the point as Cartesian:
Stored as Polar [9.961946980917455,0.8715574274765816]
```

Design 3

```
hasanalyabroudi@Hasans-MacBook-Pro pointcp % java PointCPTest
Cartesian-Polar Coordinates Conversion Program
Enter the design you want to test (Design2 / Design3 / Design5): design2

Enter the value of Rho using a decimal point(.): 10
Enter the value of Theta using a decimal point(.): 5

Testing Design 2 (PointCP2):

You entered:
Stored as Polar [10.0,5.0]


After asking to store as Polar:
Stored as Polar [10.0,5.0]

Cartesian coordinates: (9.961946980917455, 0.8715574274765816)

This is the point as Cartesian:
Stored as Polar [9.961946980917455,0.8715574274765816]
```

Design 5

```
hasanalyabroudi@Hasans-MacBook-Pro pointcp % java PointCPTest
Cartesian-Polar Coordinates Conversion Program
Enter the design you want to test (Design2 / Design3 / Design5): design5
Choose the coordinate system (P for Polar, C for Cartesian): P
Enter the value of Rho using a decimal point(.): 10
Enter the value of Theta using a decimal point(.): 5

You entered:
Polar [10.0,5.0]

After asking to store as Cartesian:
 X:9.961946980917455 Y: 0.8715574274765816
```

```
hasanalyabroudi@Hasans-MacBook-Pro pointcp % java PointCPTest
Cartesian-Polar Coordinates Conversion Program
Enter the design you want to test (Design2 / Design3 / Design5): design5
Choose the coordinate system (P for Polar, C for Cartesian): C
Enter the value of X using a decimal point(.): 10
Enter the value of Y using a decimal point(.): 5

You entered:
Cartesian (10.0,5.0)

After asking to store as Polar:
 Rho:11.180339887498949 Theta: 26.56505117707799
```