Activity #23: UML and Class Design
Recorder's Report

Manager: Reader:

Recorder: Driver:

Date: Score: Satisfactory / Not Satisfactory

Record your team's answers to the key questions (marked with ) below.

a) Model 1, Question #4

b) Model 2, Question #8

c) Model 3, Question #11

Activity #23: UML and Class Design

In this activity, you will work in teams of 3–4 students to learn new concepts. This activity will introduce you to UML and Class Design in C++.

Content Learning Objectives

After completing this activity, students should be able to:

- Explain class diagrams in UML
- Implement a class based on a UML diagram

Process Skill Goals

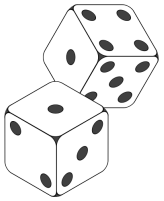
During the activity, students should make progress toward:

- Write method signatures exactly as shown in a UML diagram
- Design a new class as a UML diagram based on a general description

Model 1 The Die Class

Die
-face: int
+Die()
+getFace(): int
+roll(): int

```
1  class Die {
2    public:
3      Die() { this->face = 1; }
4      int getFace() const {
5        return this->face;
6      }
7      int roll() {
8        this->face = rand() % 6 + 1;
9        return this->face;
10     }
11   private:
12     int face;
13 };
14
```



Refer to Model 1 above as your team develops consensus answers to the questions below.

Questions (20 min) Start time:

1. In the *Unified Modeling Language* (UML), a class diagram (on the left in our model) provides a way of graphically illustrating a class's design, independent of programming language.

a) What are the attributes of `Die` and what are its methods?

b) How are attributes and methods distinguished in the class diagram?

2. Notice that several special symbols are used in the class diagram. Compare the diagram with the given C++ implementation of the class to help you answer the following questions.

a) In the class diagram, what do the "-" and "+" symbols represent?

b) What does the ":" represent?

3. The file `activity23a.cpp` contains the class from this model along with an example `main` program that utilizes it.

a) What would you change to make this a 5-sided die?

b) How would you change the class diagram to reflect this change in the code?

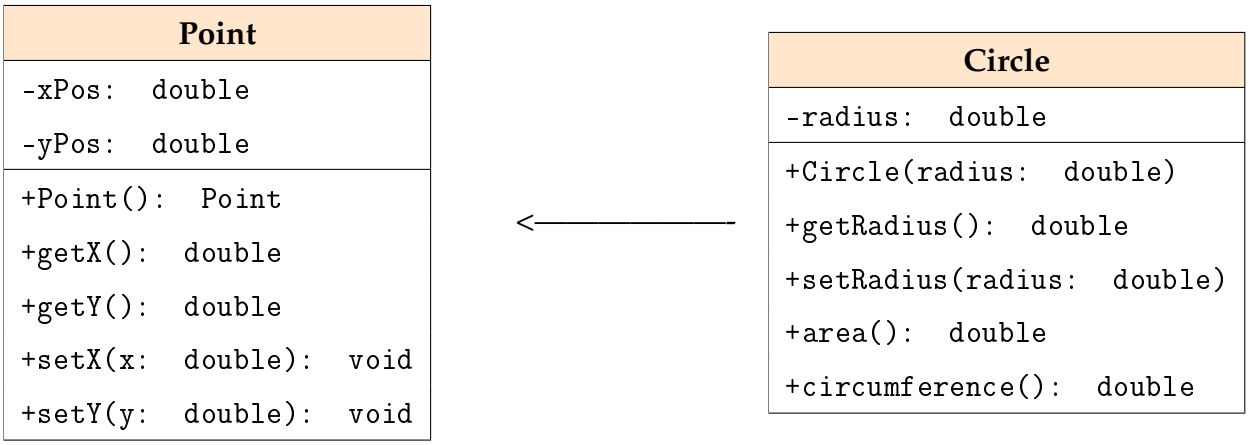
4. Suppose you wanted to generalize this class so that it could represent a die with any number of sides, set by the constructor.

a) What new attribute(s) and/or method(s) would you need to add?

b) What methods would you need to alter?

c) What would the class diagram look like after these changes?

Model 2 A Circle Class



Refer to Model 2 above as your team develops consensus answers to the questions below.

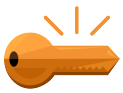
Questions (15 min) Start time:

5. What are the attributes and methods of `Circle`, and what is their visibility?

6. Based on this model and the previous one, what parts of a class are typically private, and what parts are typically public?

7. What does the arrow pointing from the `Circle` class diagram to the `Point` class diagram represent?

8. The file `activity04b.cpp` contains an implementation of the `Point` class. Write an implementation of the `Circle` class that matches this diagram.



Model 3 A Credit Card

Refer to Model 3 above as your team develops consensus answers to the questions below.

Questions (15 min) **Start time:**

9. List two or more attributes that would be necessary for a class to model a `CreditCard`. Indicate what data types would be most appropriate.

10. When constructing or updating a `CreditCard` objects, what values would you need to check? What are the valid ranges of each attribute?

11. Draw a UML class diagram for your implementation of the `CreditCard` class.

12. Many banks offer `CashBack` credit cards which accumulate a certain number of points for every dollar spent. Other banks offer `TravelRewards` cards which accumulate miles on a particular airline for every dollar spent. Construct a UML diagram with appropriate attributes and methods showing classes for these two types of cards and their relationship to the original `CreditCard` class you defined above.