CS2030S

AY24/25 S2 **Problem Set 01** 22-23 January 2025

1. *This question is adapted from the CS2030S midterm test of AY 21/22 Sem 2.*

   Consider the following Java program:

```
class BankAccount {
  double balance;

  BankAccount(double initBalance) {
    this.balance = initBalance;
  }
}

class Customer {
  BankAccount account;

  Customer() {
    this.account = new BankAccount(0);
  }

  public void deposit(double amount) {
    this.account.balance += amount;
  }

  public boolean withdraw(double amount) {
    if (this.account.balance >= amount) {
      this.account.balance -= amount;
      return true;
    }
    return false;
  }
}
```

   (a) Does this program follow the principle of information hiding? Explain.

   (b) Does this program follow the principle of "Tell, Don't Ask?" Explain.

   (c) If you think the program violates any of the principles in Parts (a) and (b), revise the program so that it adheres to the principles.

2. Consider the following definition of a `Vector2D` class:

```
class Vector2D {
  private double x;
  private double y;

  public Vector2D(double x, double y) {
    this.x = x;
    this.y = y;
  }

  public void add(Vector2D v) {
    this.x = this.x + v.x;
    this.y = this.y + v.y;
    // line A
  }
}
```

   Suppose that the following program fragment is in a `main` method,

```
Vector2D v1 = new Vector2D(1, 1);
Vector2D v2 = new Vector2D(2, 2);
v1.add(v2);
```

(a) Show the content of the stack and the heap when the execution reaches the line labeled `A` above. Label your variables and the values they hold clearly. You can use arrows to indicate object references. Draw boxes around the stack frames of the methods `main` and `add`, and label them.

(b) Suppose that the representation of `x` and `y` have been changed to a `double` array:

```
class Vector2D {
  private double[] coord2D;
  :
}
```

What changes do you need for the other parts of class `Vector2D`?

Would the program fragment above still be valid?

3. Study the following `Point` and `Circle` classes.

```
public class Point {
  private double x;
  private double y;

  public Point(double x, double y) {
    this.x = x;
    this.y = y;
  }
}

public class Circle {

  private Point centre;
  private int radius;

  public Circle(Point centre, int radius) {
    this.centre = centre;
    this.radius = radius;
  }

  @Override
  public boolean equals(Object obj) {
    System.out.println("equals(Object) called");
    if (obj == this) {
      return true;
    }
    if (obj instanceof Circle) {
      Circle circle = (Circle) obj;
      return (circle.centre.equals(centre) && circle.radius == radius);
    } else {
      return false;
    }
  }

  public boolean equals(Circle circle) {
    System.out.println("equals(Circle) called");
    return circle.centre.equals(centre) && circle.radius == radius;
```

```
  }
}
```

Given the following program fragment,

```
Circle c1 = new Circle(new Point(0, 0), 10);
Circle c2 = new Circle(new Point(0, 0), 10);
Object o1 = c1;
Object o2 = c2;
```

(a) What is the return value of `c1.equals(c2)`? Explain.

(b) For each of the statement below, trace through the two-step dynamic binding process to show which `equals` method is invoked during run-time.

(i) `o1.equals(o2);`

(ii) `o1.equals((Circle) o2);`

(iii) `o1.equals(c2);`

(iv) `c1.equals(o2);`

(v) `c1.equals((Circle) o2);`

(vi) `c1.equals(c2);`

Homework

4. In this question, your task is to create an abstraction for a single-digit ternary number, that can only store the values 0, 1, or 2.

(a) Write a class called `Ternary` with an `int` field named `value`. The field should not be accessible from outside the class. The class should have a constructor that initializes `value` to 0, and a `toString` method that returns the `value` as a `String`.

Example of how the class can be used:

jshell> Ternary t = new Ternary(); t ==> 0

Note: You can use the static method `String::valueOf` to convert an `int` to a `String`. See the Java API for `String` for more information.

(b) Add a method called `incr` to the class. `incr` should increment `value` by one but wraps around to 0 when the value exceeds 2. The method should not return anything.

Example of how the class can be used:

jshell> Ternary t = new Ternary(); t ==> 0

jshell> t.incr() jshell> t t ==> 1

jshell> t.incr() jshell> t t ==> 2

jshell> t.incr() jshell> t t ==> 0