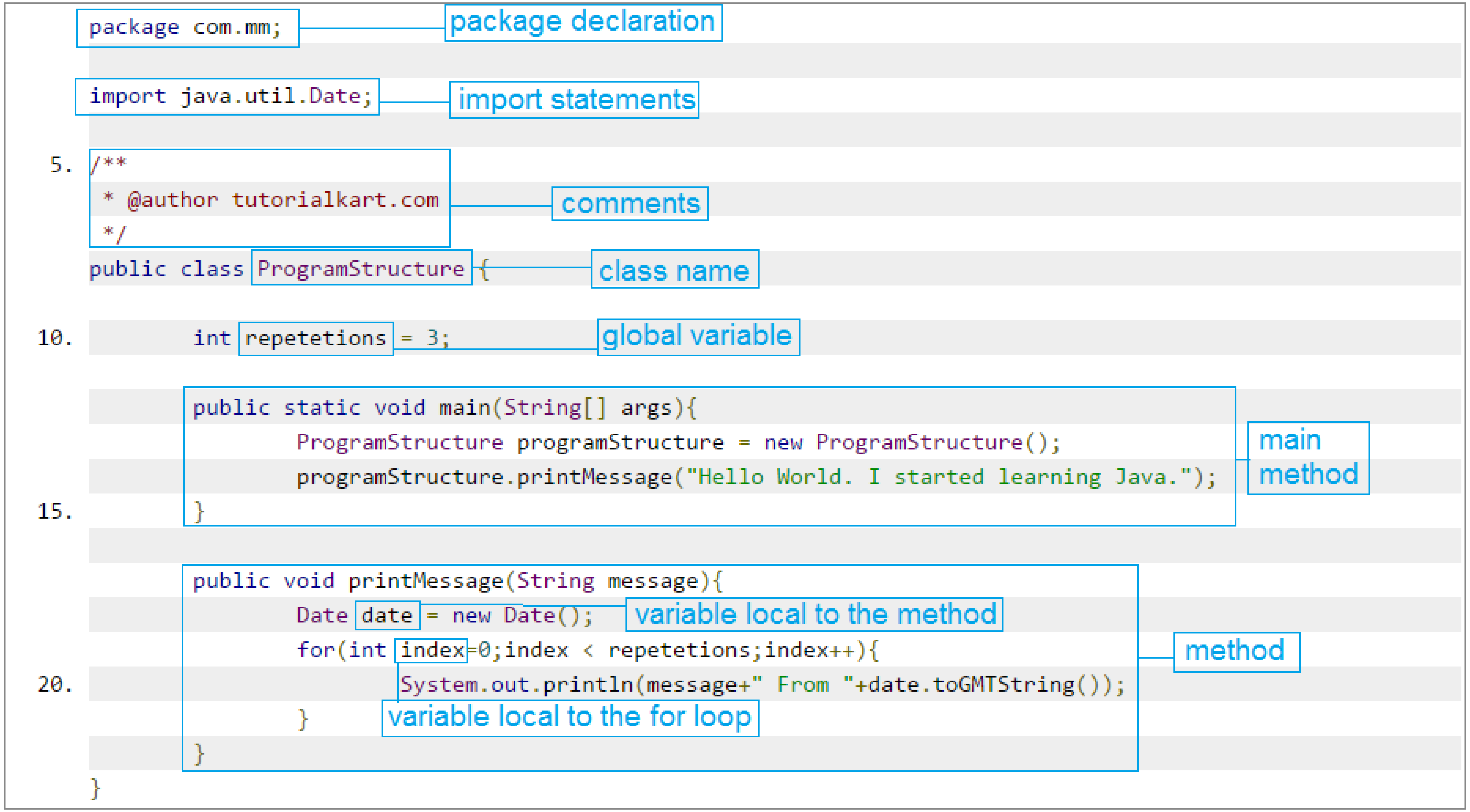
# TUTORIAL 2

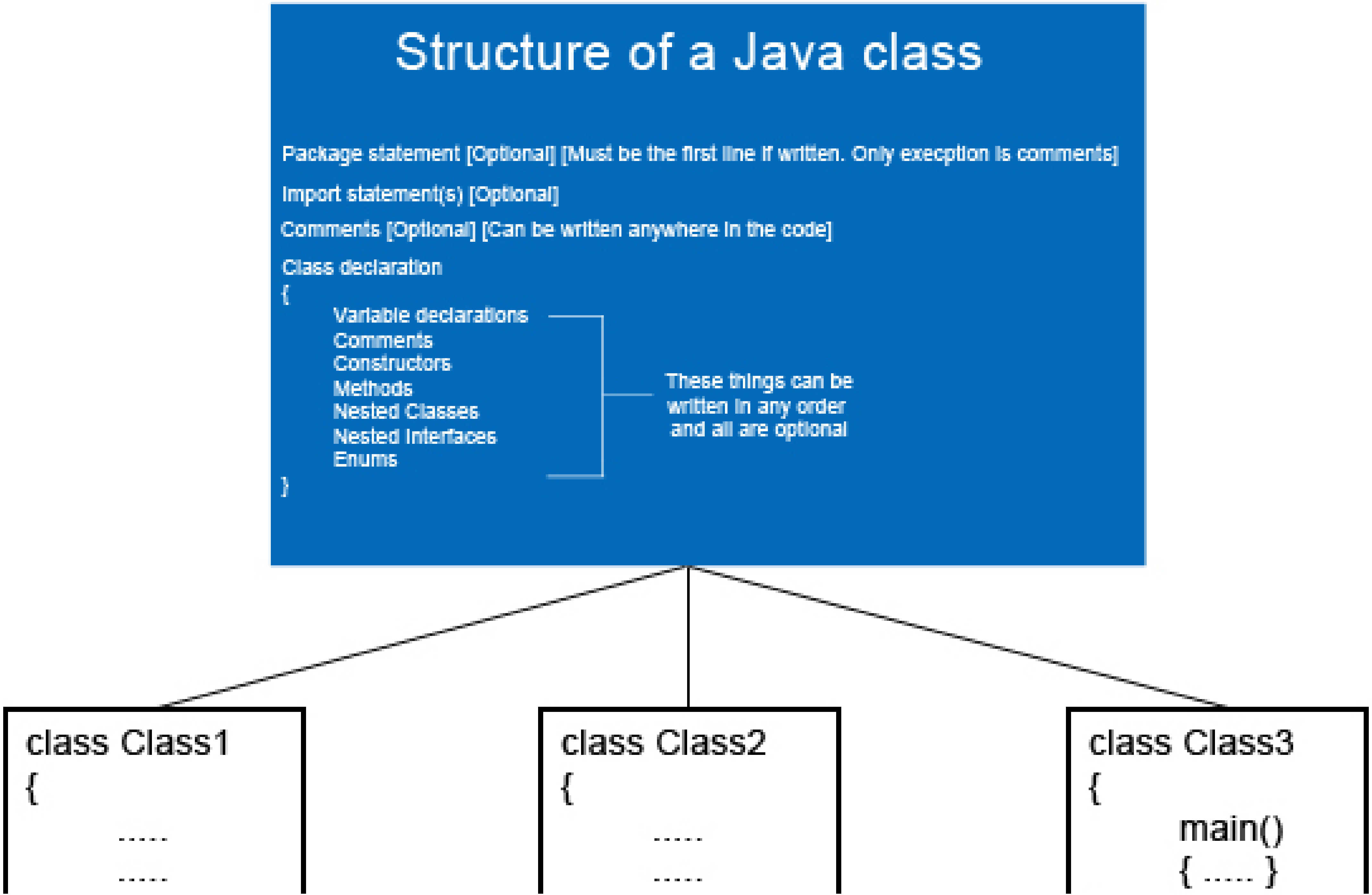
Classes & Objects

## Using multiple classes in a Java program



```java
package com.mm;

import java.util.Date;

/**
 * @author tutorialkart.com
 */
public class ProgramStructure {

	int repetetions = 3;

	public static void main(String[] args){
		ProgramStructure programStructure = new ProgramStructure();
		programStructure.printMessage("Hello World. I started learning Java.");
	}

	public void printMessage(String message){
		Date date = new Date();
		for(int index=0;index < repetetions;index++){
			System.out.println(message+" From "+date.toGMTString());
		}
	}
}
```

https://www.tutorialkart.com/java/structure-of-a-java-program/



# Structure of a Java class

Package statement [Optional] [Must be the first line if written. Only execption is comments]

Import statement(s) [Optional]

Comments [Optional] [Can be written anywhere in the code]

```
Class declaration
{
        Variable declarations
        Comments
        Constructors
        Methods
        Nested Classes
        Nested Interfaces
        Enums
}
```

These things can be written in any order and all are optional

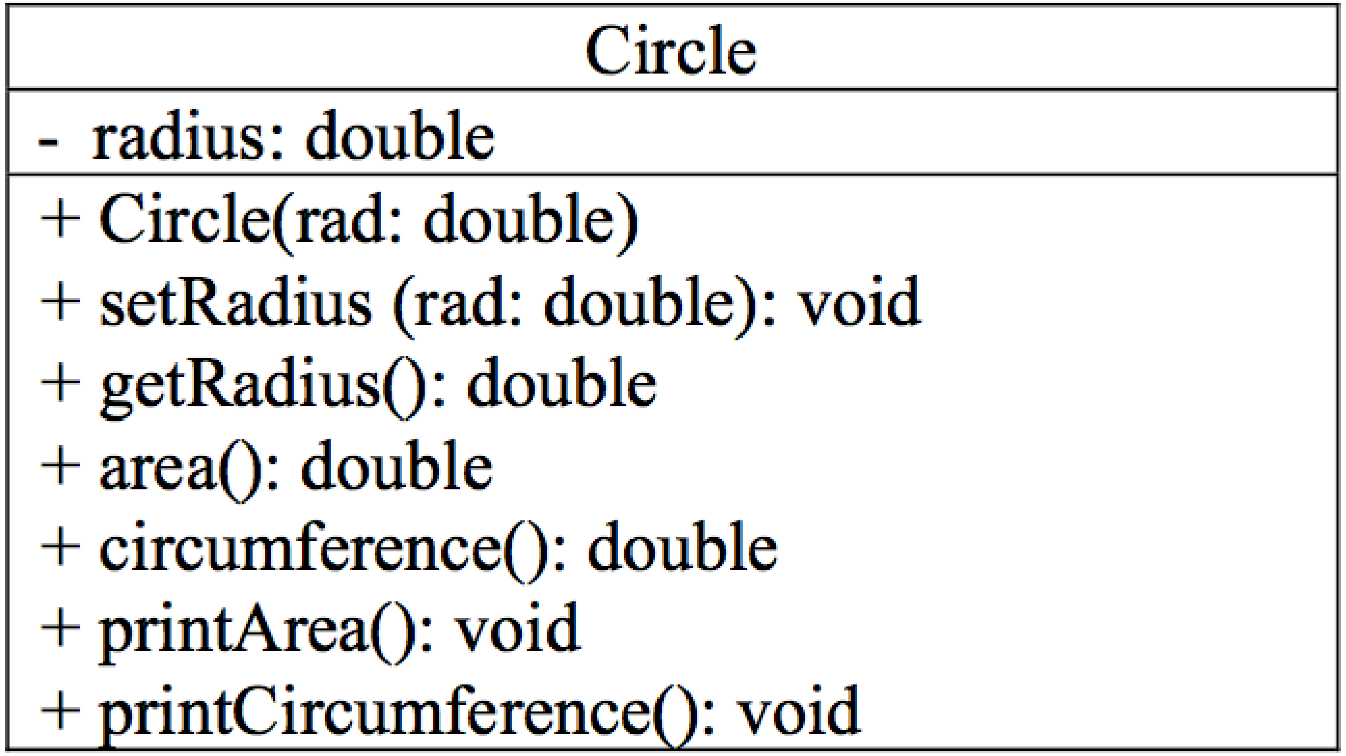
```
class Class1
{
        .....
        .....
```

```
class Class2
{
        .....
        .....
```

```
class Class3
{
        main()
        { ..... }
```

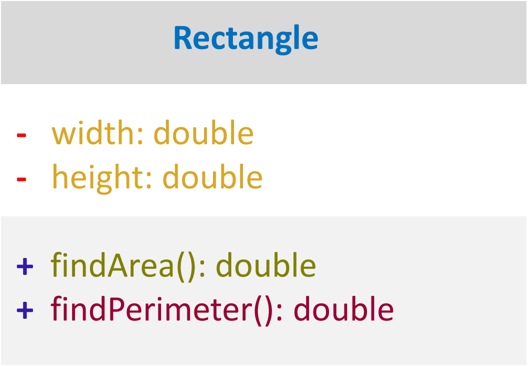
# Q1

- The UML class diagram for the Circle class is given below:



```
Circle
- radius: double
+ Circle(rad: double)
+ setRadius (rad: double): void
+ getRadius(): double
+ area(): double
+ circumference(): double
+ printArea(): void
+ printCircumference(): void
```

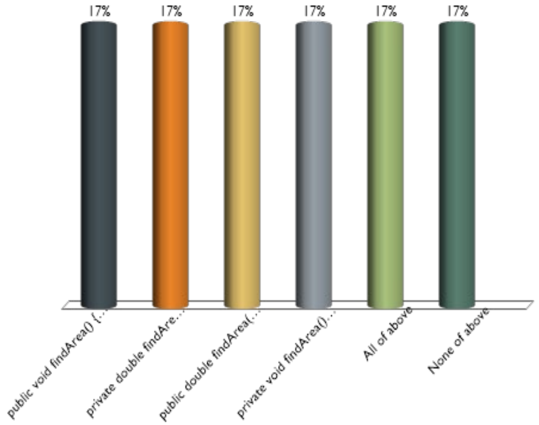
# GIVEN THE UML DIAGRAM OF RECTANGLE, WHICH OF THE FOLLOWING IS THE CORRECT CODE FOR METHOD FINDAREA?



**Rectangle**

- width: double
- height: double

+ findArea(): double
+ findPerimeter(): double



A. public void findArea() { return width * height ;}

B. private double findArea() { return width * height ;}

C. public double findArea() { width * height ; }

D. private void findArea() { return width * height ;}

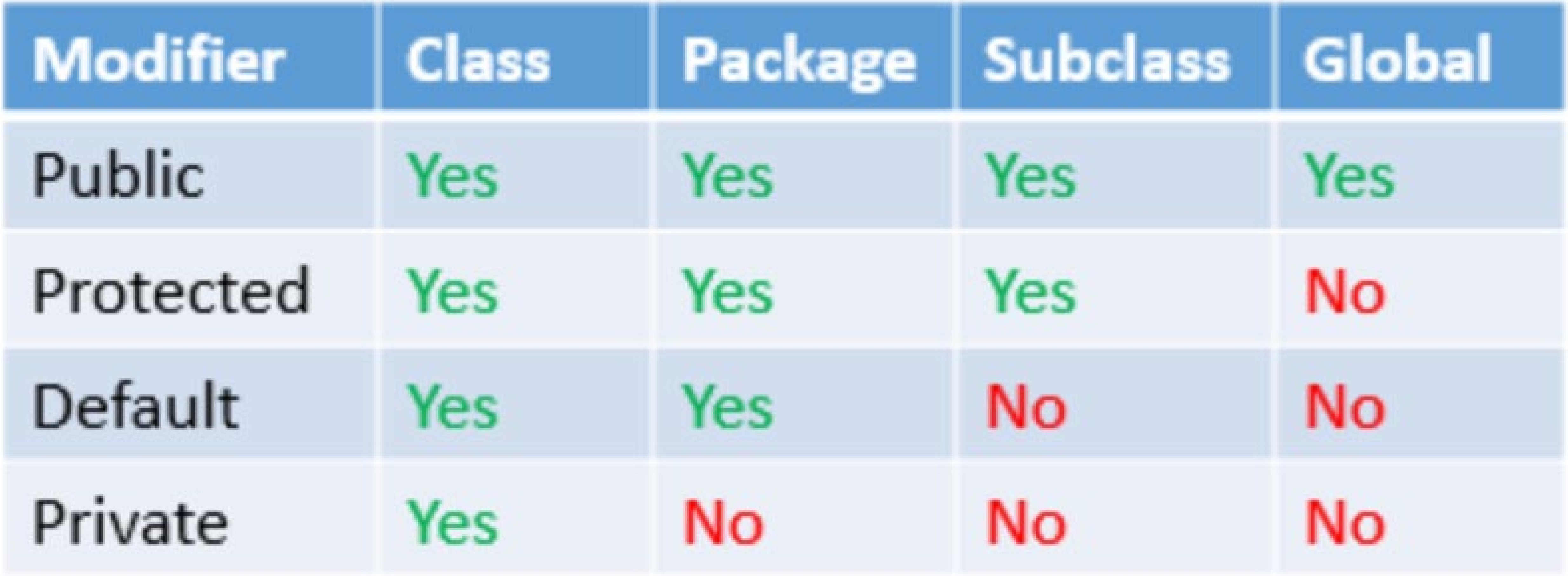
E. All of above

✓ F. None of above

# Q1

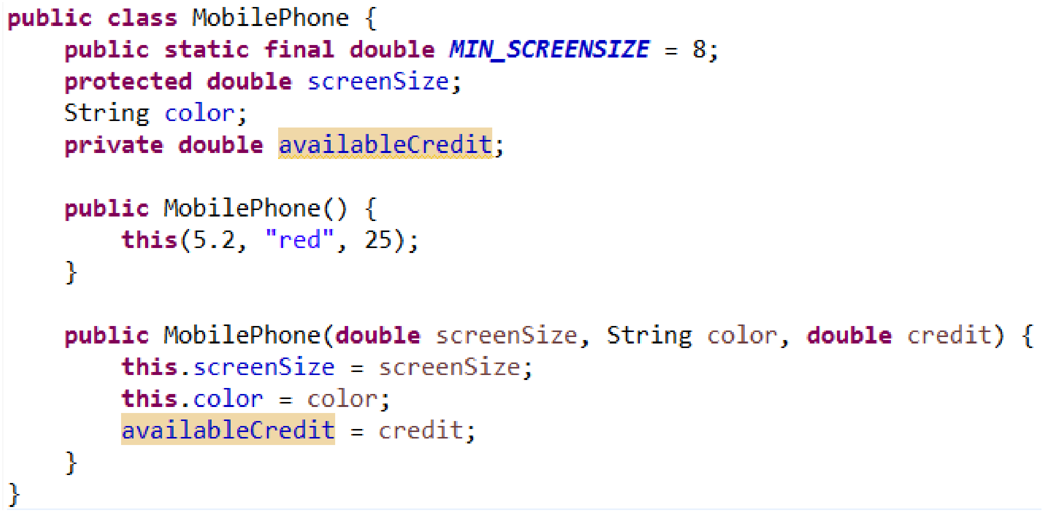
- Write a class **Circle** that has the following instance variables and methods:

```
public class Circle {
   private double radius;          // radius of circle
   private static final double PI = 3.14159;
   public Circle(double rad) {...} // constructor
   public void setRadius(double rad){...} // mutator
   public double getRadius(){...} // accessor method
   public double area(){...} // calculate area
   public double circumference() {...}
   public void printArea(){...} // print area
   public void printCircumference(){...} }
```

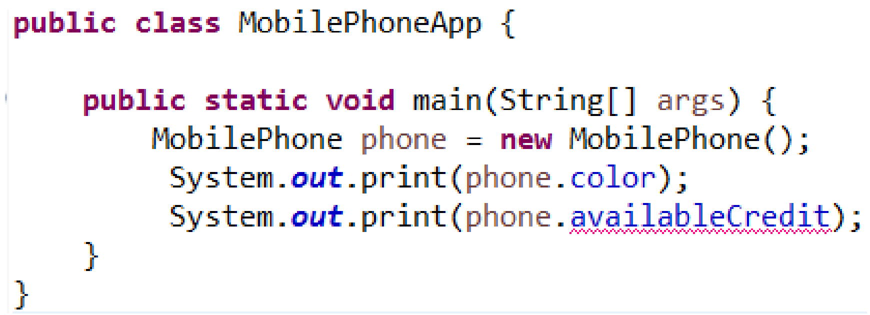


| Modifier | Class | Package | Subclass | Global |
|---|---|---|---|---|
| Public | Yes | Yes | Yes | Yes |
| Protected | Yes | Yes | Yes | No |
| Default | Yes | Yes | No | No |
| Private | Yes | No | No | No |

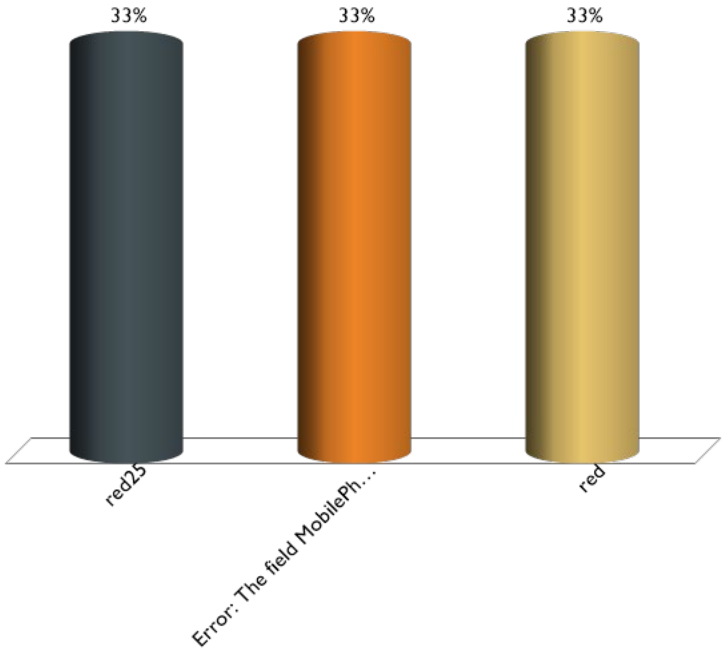
# WHAT IS THE OUTPUT OF THE FOLLOWING PROGRAM?



```java
public class MobilePhone {
    public static final double MIN_SCREENSIZE = 8;
    protected double screenSize;
    String color;
    private double availableCredit;

    public MobilePhone() {
        this(5.2, "red", 25);
    }

    public MobilePhone(double screenSize, String color, double credit) {
        this.screenSize = screenSize;
        this.color = color;
        availableCredit = credit;
    }
}
```



```java
public class MobilePhoneApp {

    public static void main(String[] args) {
        MobilePhone phone = new MobilePhone();
         System.out.print(phone.color);
         System.out.print(phone.availableCredit);
    }
}
```

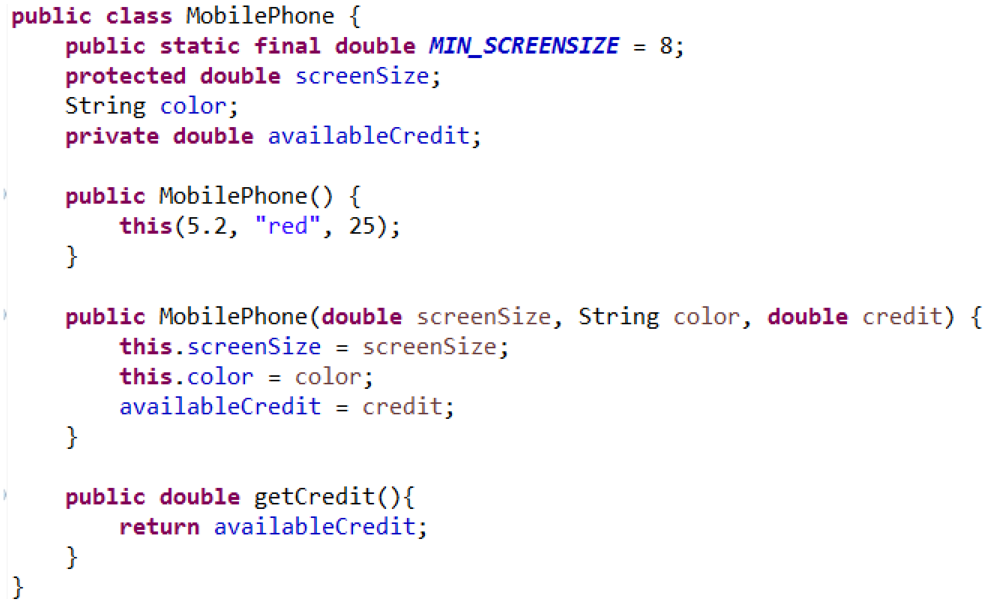


A. red25.0

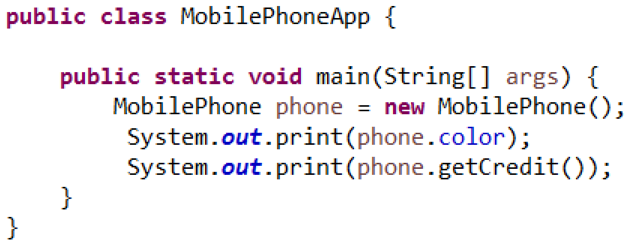
✓ B. Error: The field MobilePhone.availableCredit is not visible

C. red

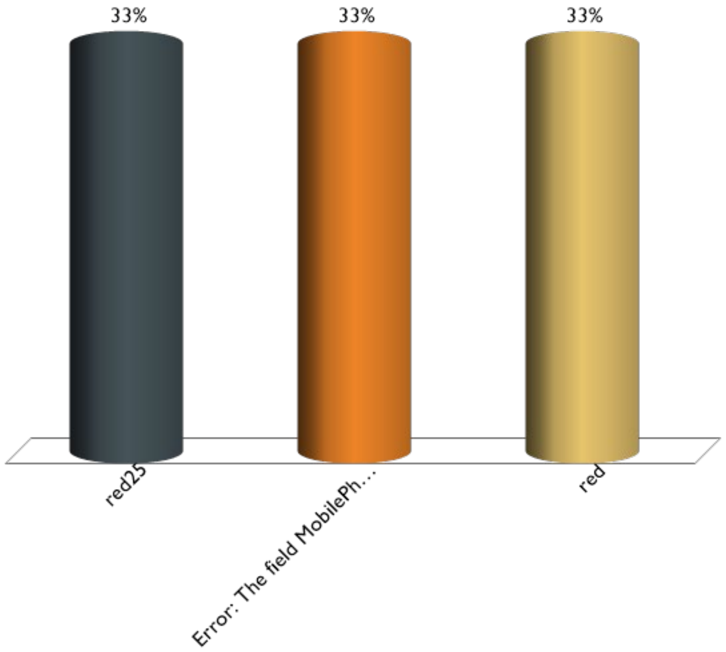
# WHAT IS THE OUTPUT OF THE FOLLOWING PROGRAM?



```java
public class MobilePhone {
    public static final double MIN_SCREENSIZE = 8;
    protected double screenSize;
    String color;
    private double availableCredit;

    public MobilePhone() {
        this(5.2, "red", 25);
    }

    public MobilePhone(double screenSize, String color, double credit) {
        this.screenSize = screenSize;
        this.color = color;
        availableCredit = credit;
    }

    public double getCredit(){
        return availableCredit;
    }
}
```



```java
public class MobilePhoneApp {

    public static void main(String[] args) {
        MobilePhone phone = new MobilePhone();
         System.out.print(phone.color);
         System.out.print(phone.getCredit());
    }
}
```



✓ A. red25.0

B. Error: The field MobilePhone.availableCredit is not visible

C. red

# THE DIFFERENCES BETWEEN A CONSTRUCTOR AND A METHOD ARE

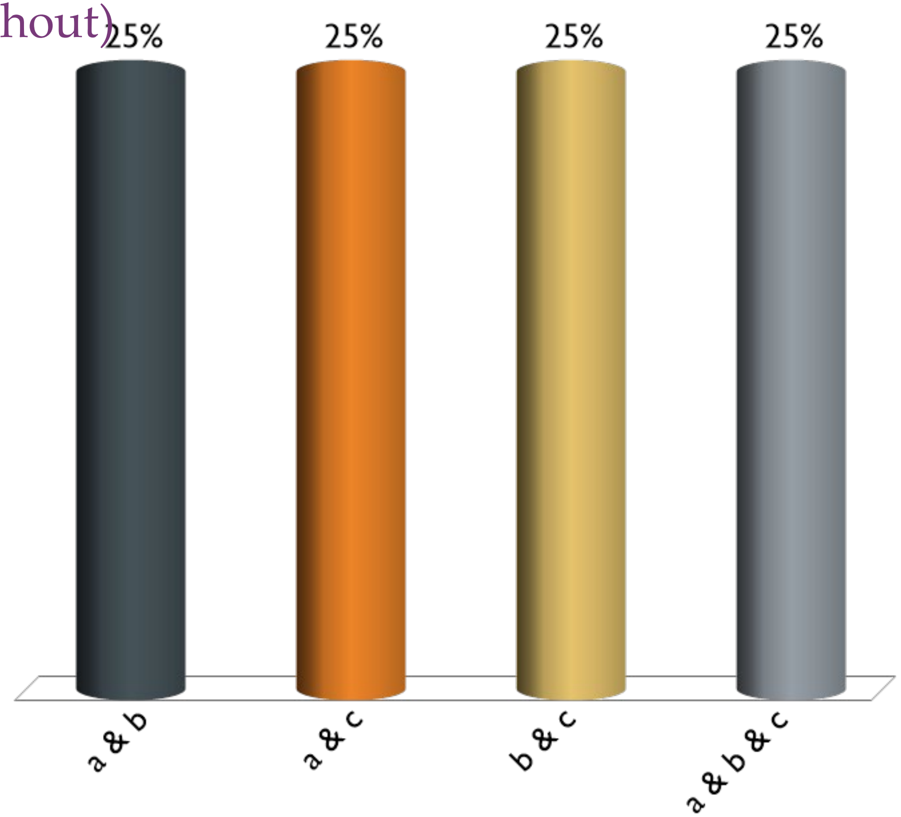
a. the number of arguments
b. the return type (with and without)
c. the naming constraint

A. a & b

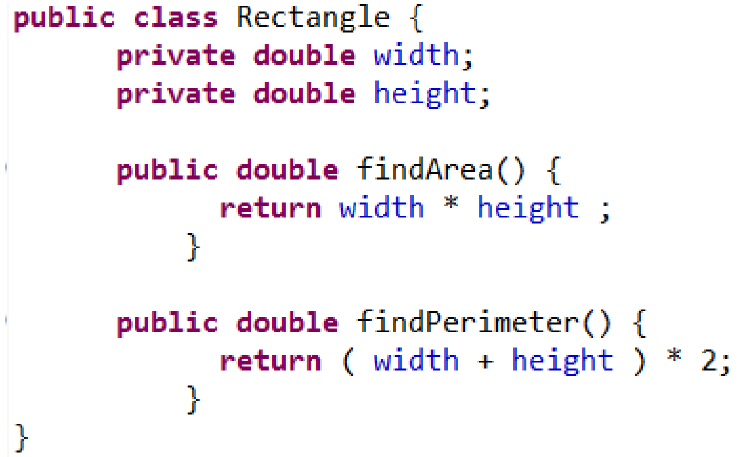
B. a & c

[✓ C. b & c]

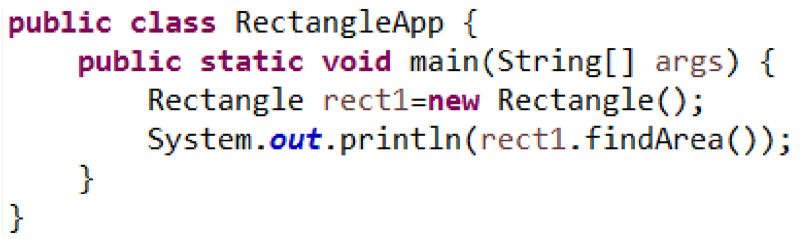
D. a & b & c



# WHAT IS THE OUTPUT OF THE FOLLOWING CODE?



```
public class Rectangle {
      private double width;
      private double height;

      public double findArea() {
            return width * height ;
         }

      public double findPerimeter() {
            return ( width + height ) * 2;
         }
}
```



```
public class RectangleApp {
    public static void main(String[] args) {
        Rectangle rect1=new Rectangle();
        System.out.println(rect1.findArea());
    }
}
```

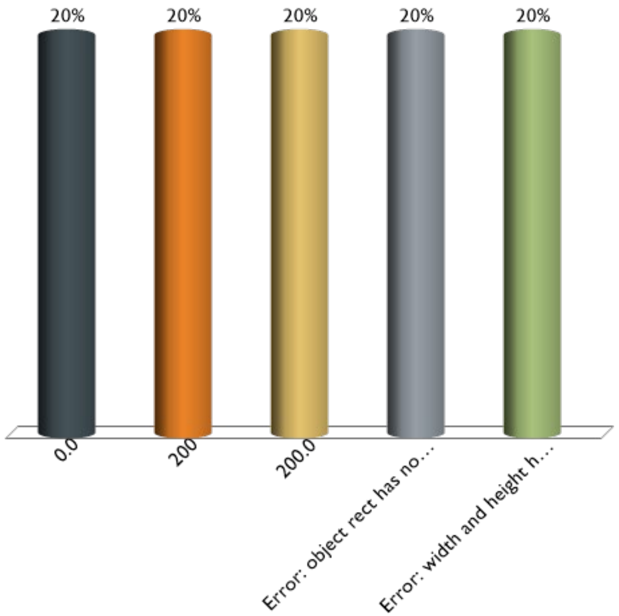
✓ A. 0.0

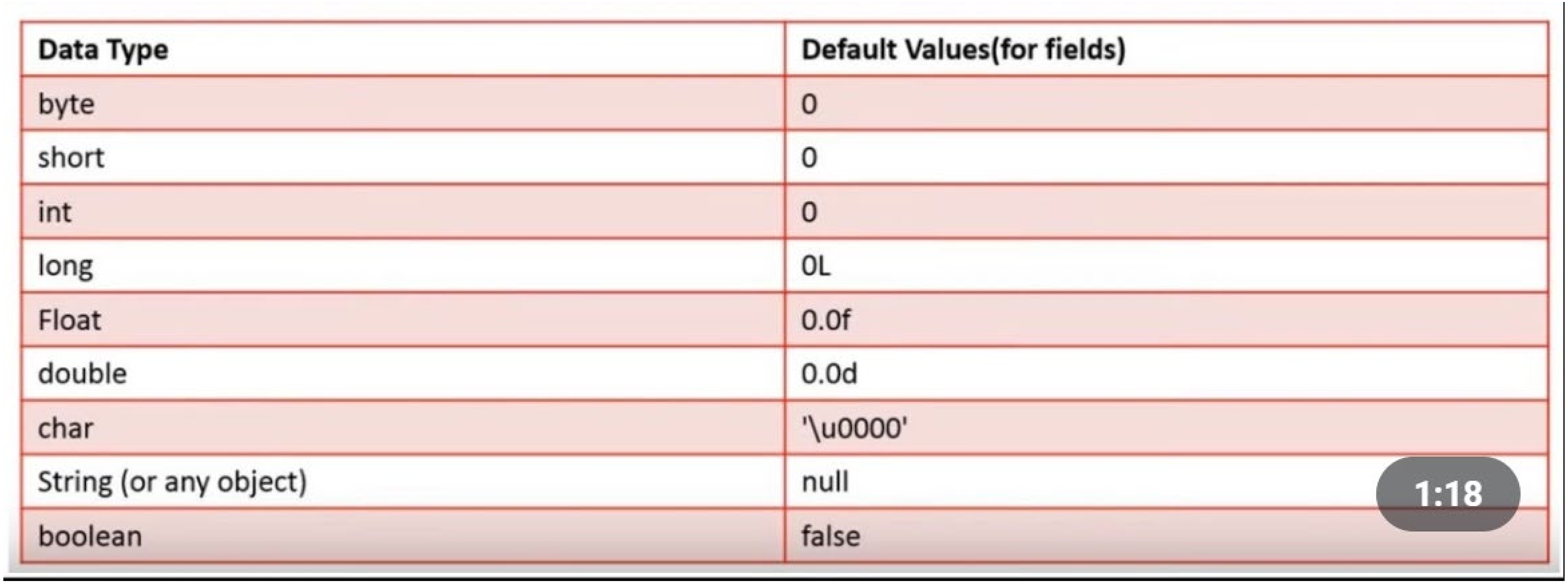
B. 200

C. 200.0

D. Error: object *rect* has not been created

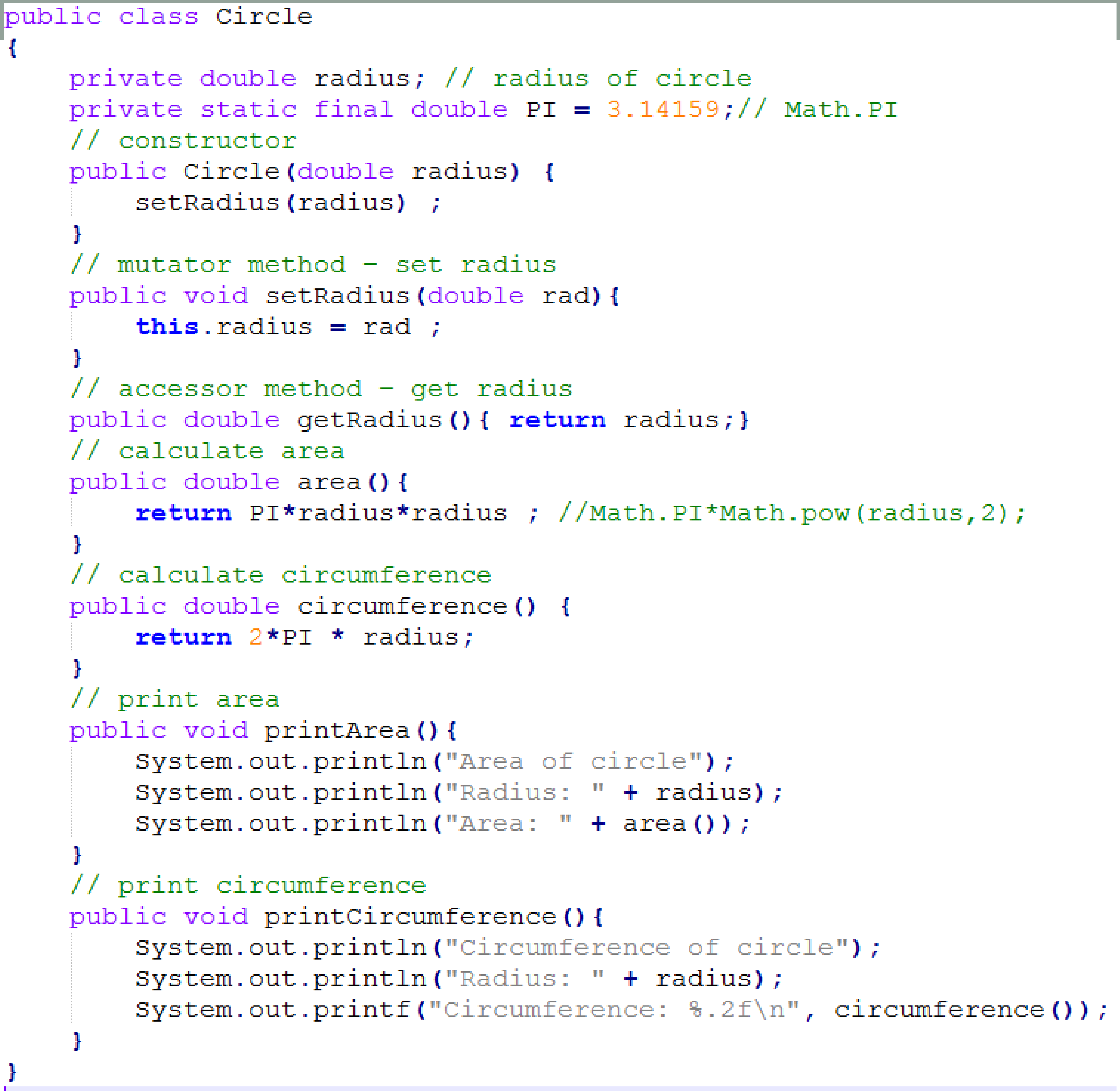
E. Error: width and height have not been initialized





| Data Type | Default Values(for fields) |
|---|---|
| byte | 0 |
| short | 0 |
| int | 0 |
| long | 0L |
| Float | 0.0f |
| double | 0.0d |
| char | '\u0000' |
| String (or any object) | null |
| boolean | false |

A1



```
public class Circle
{
    private double radius; // radius of circle
    private static final double PI = 3.14159;// Math.PI
    // constructor
    public Circle(double radius) {
        setRadius(radius) ;
    }
    // mutator method - set radius
    public void setRadius(double rad){
        this.radius = rad ;
    }
    // accessor method - get radius
    public double getRadius(){ return radius;}
    // calculate area
    public double area(){
        return PI*radius*radius ; //Math.PI*Math.pow(radius,2);
    }
    // calculate circumference
    public double circumference() {
        return 2*PI * radius;
    }
    // print area
    public void printArea(){
        System.out.println("Area of circle");
        System.out.println("Radius: " + radius);
        System.out.println("Area: " + area());
    }
    // print circumference
    public void printCircumference(){
        System.out.println("Circumference of circle");
        System.out.println("Radius: " + radius);
        System.out.printf("Circumference: %.2f\n", circumference());
    }
}
```

# Q1

- Write an application class `CircleApp` to test the `Circle` class.
- The class `CircleApp` should display a menu.
- The user can then select an option of the following:

  (1) create a new circle

  (2) print area

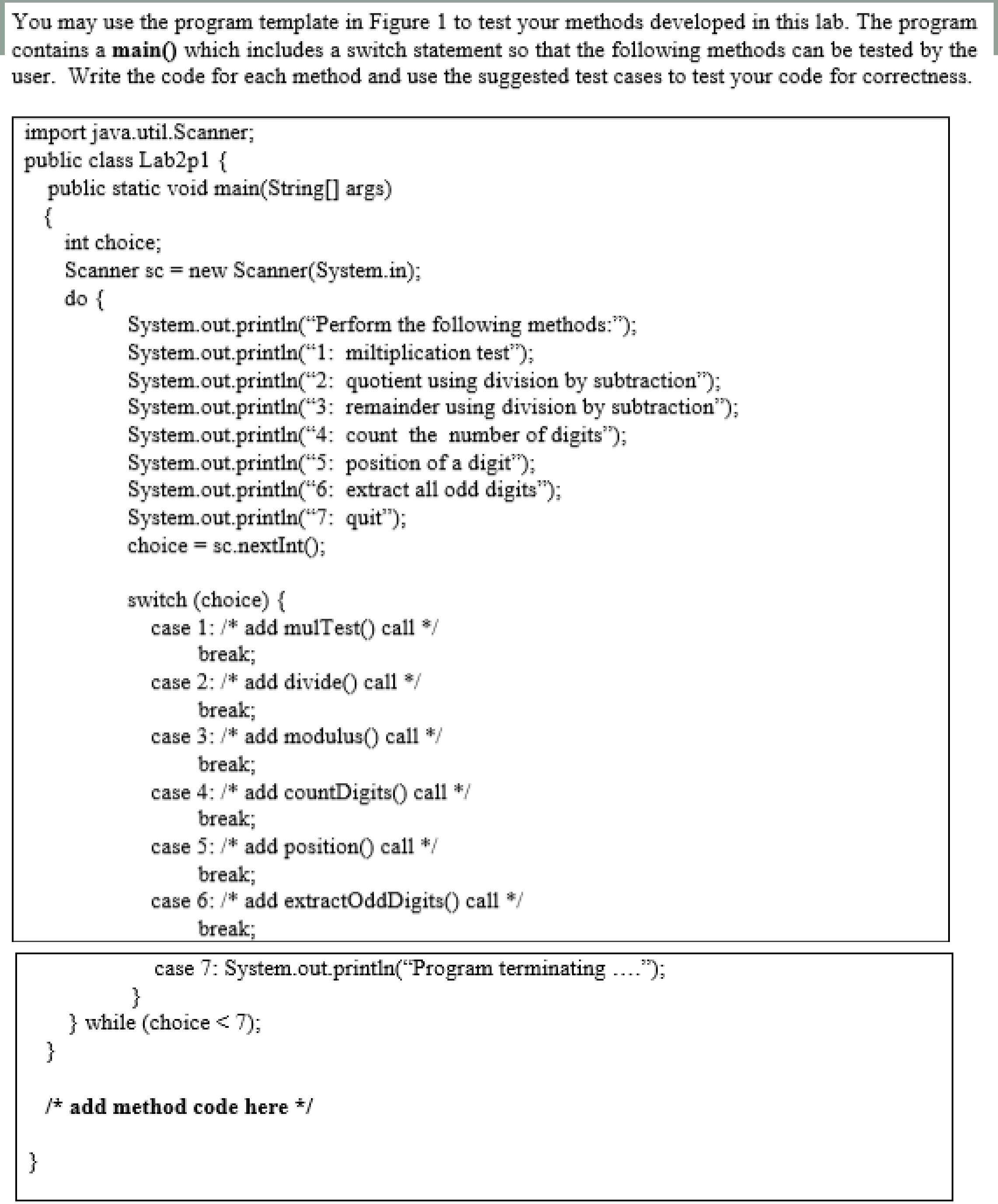
  (3) print circumference

  (4) quit
- Implement the operations for each option.

# Q1

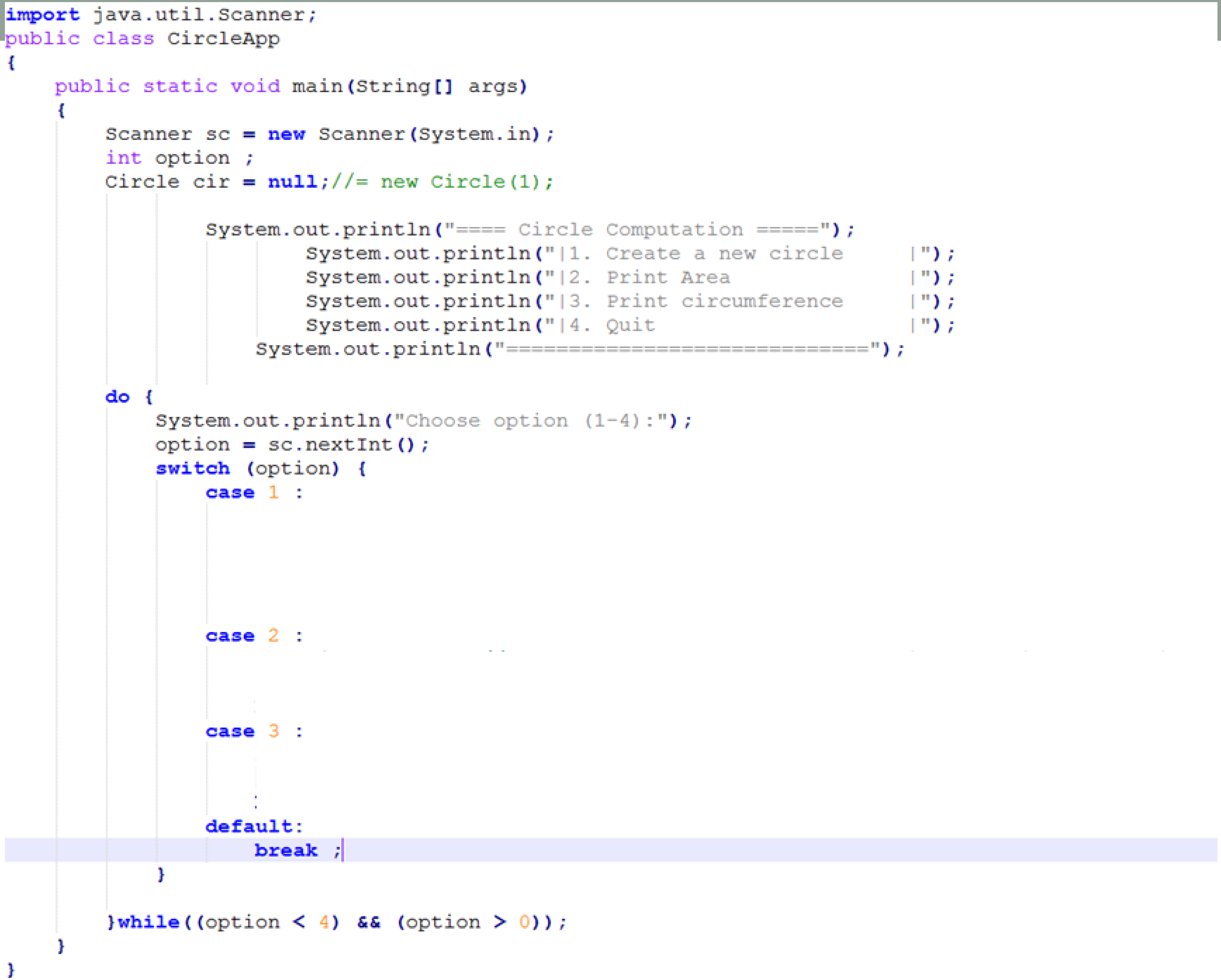
```
==== Circle Computation =====
|1. Create a new circle       |
|2. Print Area                |
|3. Print circumference       |
|4. Quit                      |
==============================
Choose option (1-3):
1
Enter the radius to compute the area and circumference
5
A new circle is created
Choose option (1-3):
2
Area of circle
Radius: 5.0
Area: 78.53975
Choose option (1-3):
3
Circumference of circle
Radius: 5.0
Circumference: 31.4159
Choose option (1-3):
4
Thank you!!
```

# Program template for Lab 2.

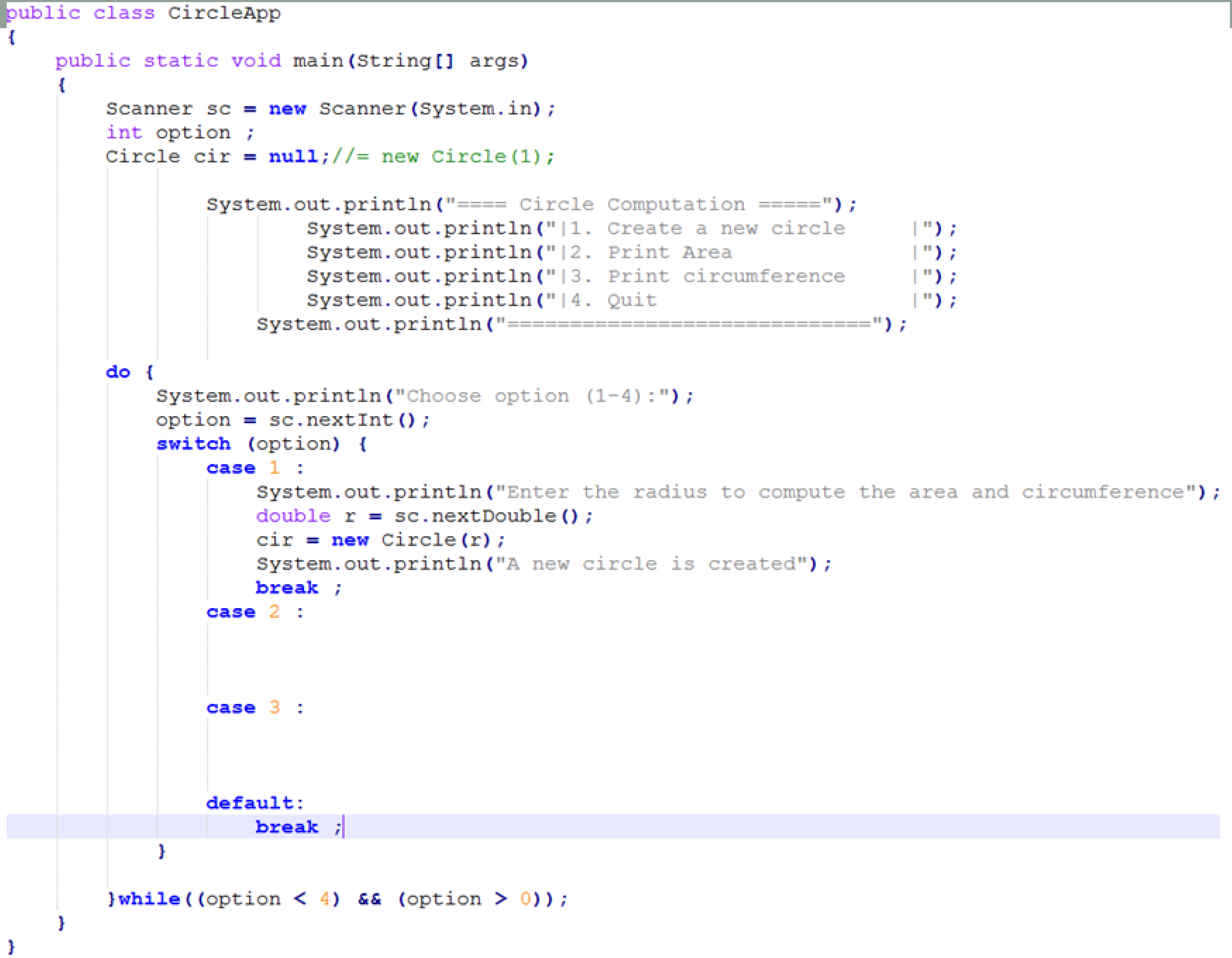


You may use the program template in Figure 1 to test your methods developed in this lab. The program contains a **main()** which includes a switch statement so that the following methods can be tested by the user. Write the code for each method and use the suggested test cases to test your code for correctness.

```
import java.util.Scanner;
public class Lab2p1 {
   public static void main(String[] args)
   {
      int choice;
      Scanner sc = new Scanner(System.in);
      do {
              System.out.println("Perform the following methods:");
              System.out.println("1:  miltiplication test");
              System.out.println("2:  quotient using division by subtraction");
              System.out.println("3:  remainder using division by subtraction");
              System.out.println("4:  count  the  number of digits");
              System.out.println("5:  position of a digit");
              System.out.println("6:  extract all odd digits");
              System.out.println("7:  quit");
              choice = sc.nextInt();

              switch (choice) {
                 case 1: /* add mulTest() call */
                       break;
                 case 2: /* add divide() call */
                       break;
                 case 3: /* add modulus() call */
                       break;
                 case 4: /* add countDigits() call */
                       break;
                 case 5: /* add position() call */
                       break;
                 case 6: /* add extractOddDigits() call */
                       break;
                 case 7: System.out.println("Program terminating ….");
              }
      } while (choice < 7);
   }

   /* add method code here */

}
```



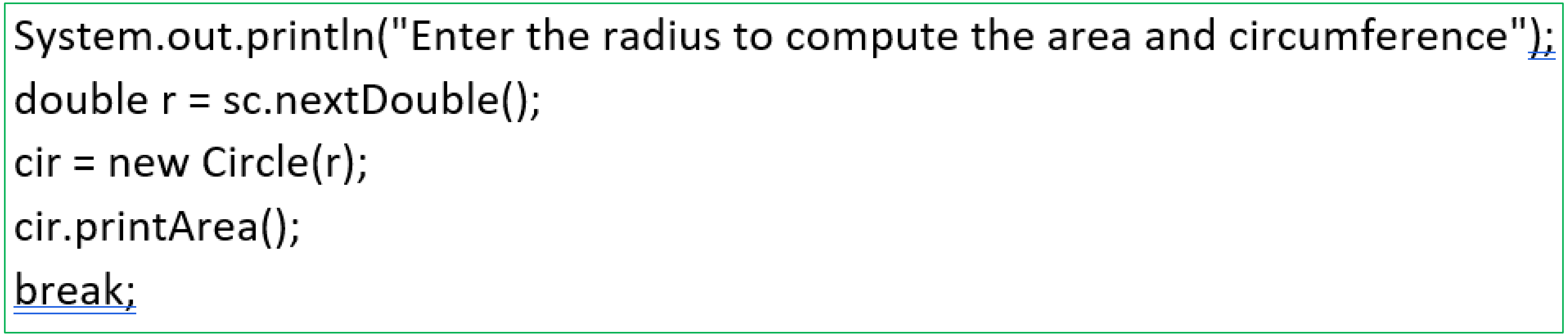
```java
import java.util.Scanner;
public class CircleApp
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        int option ;
        Circle cir = null;//= new Circle(1);

                System.out.println("==== Circle Computation =====");
                        System.out.println("|1. Create a new circle     |");
                        System.out.println("|2. Print Area              |");
                        System.out.println("|3. Print circumference     |");
                        System.out.println("|4. Quit                    |");
                    System.out.println("=============================");

        do {
            System.out.println("Choose option (1-4):");
            option = sc.nextInt();
            switch (option) {
                case 1 :




                case 2 :



                case 3 :



                default:
                    break ;
            }

        }while((option < 4) && (option > 0));
    }
}
```



```
public class CircleApp
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        int option ;
        Circle cir = null;//= new Circle(1);

                System.out.println("==== Circle Computation =====");
                        System.out.println("|1. Create a new circle     |");
                        System.out.println("|2. Print Area              |");
                        System.out.println("|3. Print circumference     |");
                        System.out.println("|4. Quit                    |");
                    System.out.println("=============================");

        do {
            System.out.println("Choose option (1-4):");
            option = sc.nextInt();
            switch (option) {
                case 1 :
                    System.out.println("Enter the radius to compute the area and circumference");
                    double r = sc.nextDouble();
                    cir = new Circle(r);
                    System.out.println("A new circle is created");
                    break ;
                case 2 :



                case 3 :



                default:
                    break ;
            }

        }while((option < 4) && (option > 0));
    }
}
```

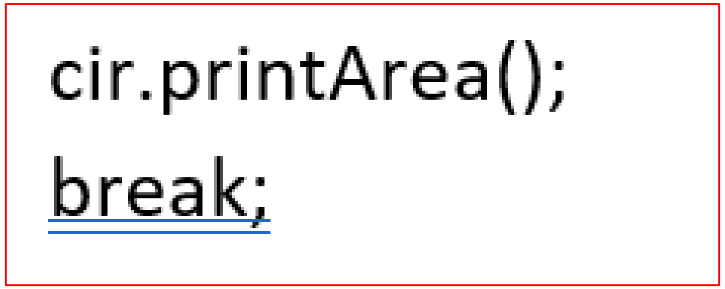
# Which of the following is the correct code segment for case 2?

A.



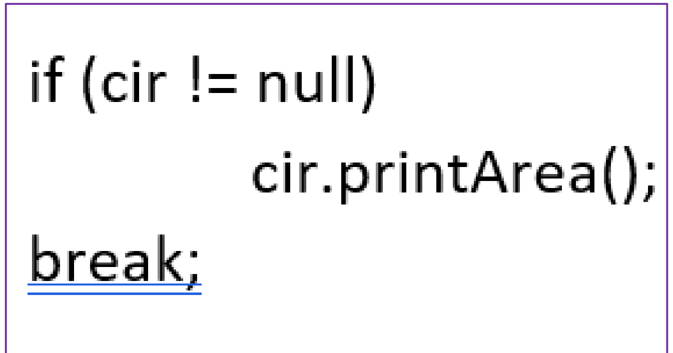
```
System.out.println("Enter the radius to compute the area and circumference");
double r = sc.nextDouble();
cir = new Circle(r);
cir.printArea();
break;
```

B.

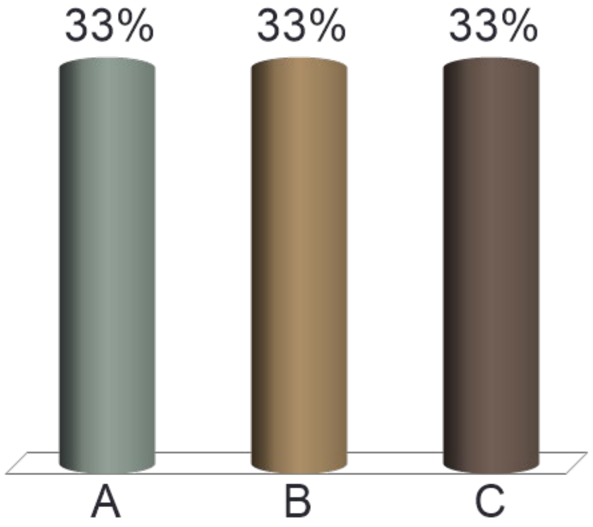


```
cir.printArea();
break;
```

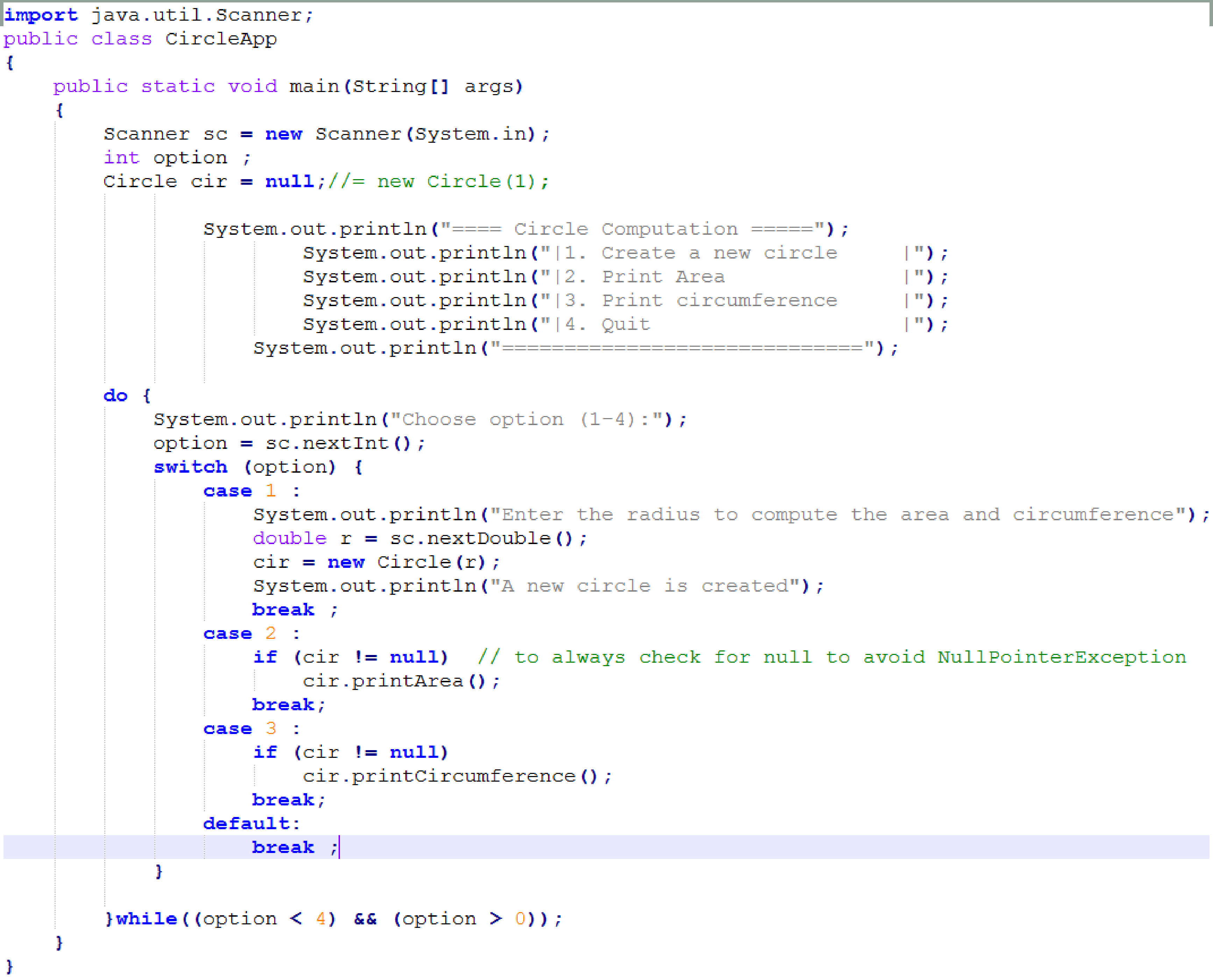
✓ C.



```
if (cir != null)
        cir.printArea();
break;
```



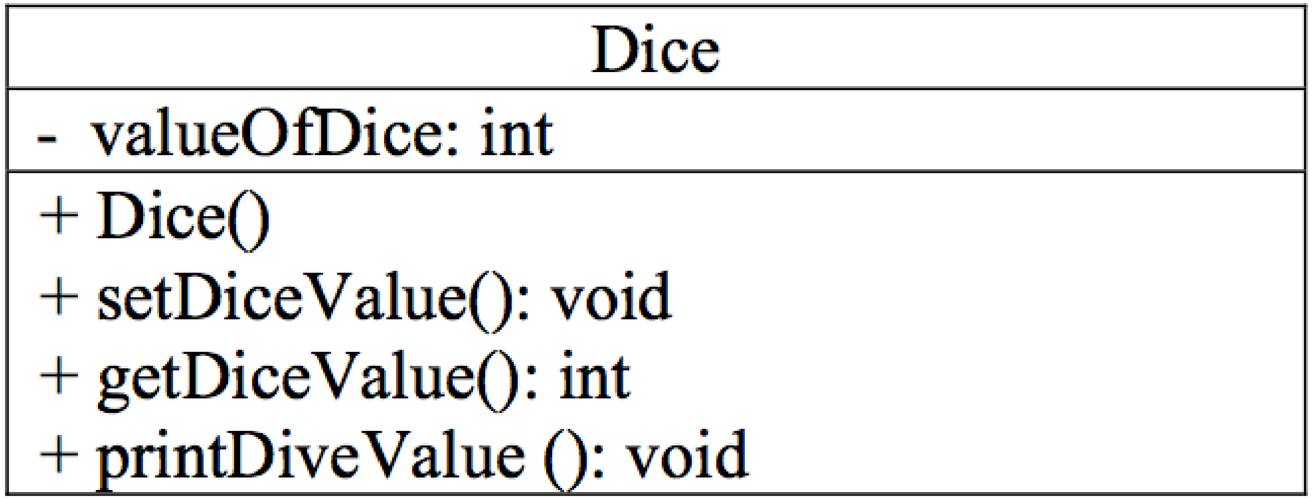
A1



```java
import java.util.Scanner;
public class CircleApp
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        int option ;
        Circle cir = null;//= new Circle(1);

                System.out.println("==== Circle Computation =====");
                        System.out.println("|1. Create a new circle     |");
                        System.out.println("|2. Print Area              |");
                        System.out.println("|3. Print circumference     |");
                        System.out.println("|4. Quit                    |");
                    System.out.println("=============================");

        do {
            System.out.println("Choose option (1-4):");
            option = sc.nextInt();
            switch (option) {
                case 1 :
                    System.out.println("Enter the radius to compute the area and circumference");
                    double r = sc.nextDouble();
                    cir = new Circle(r);
                    System.out.println("A new circle is created");
                    break ;
                case 2 :
                    if (cir != null)  // to always check for null to avoid NullPointerException
                        cir.printArea();
                    break;
                case 3 :
                    if (cir != null)
                        cir.printCircumference();
                    break;
                default:
                    break ;
            }

        }while((option < 4) && (option > 0));
    }
}
```

# Q2

- Write a class `Dice` that has the following instance variables and methods:
- The UML class diagram for the `Dice` class is given below:



Dice

- valueOfDice: int

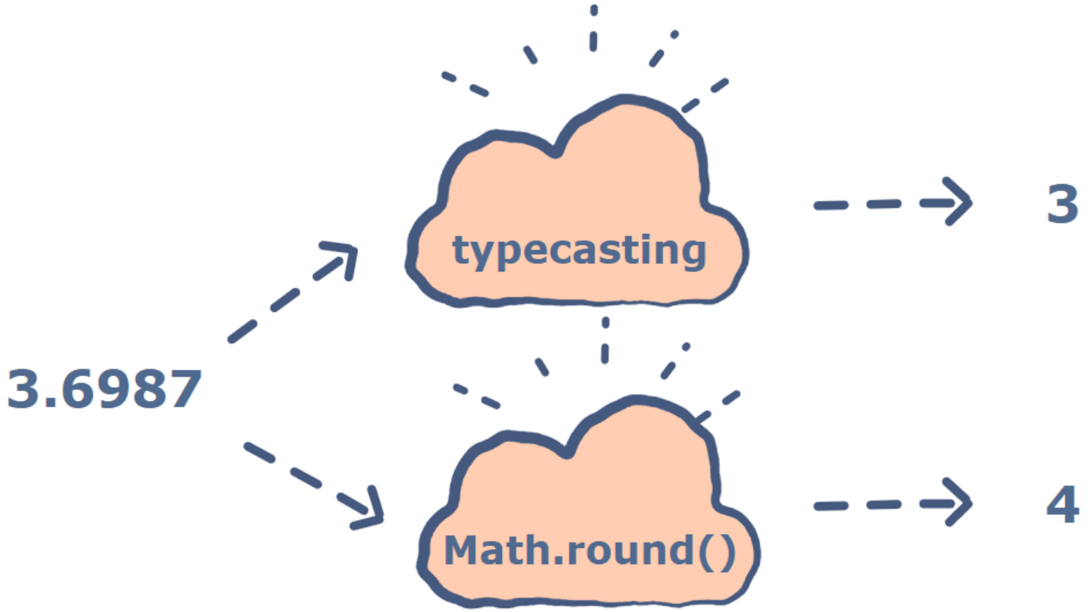
+ Dice()
+ setDiceValue(): void
+ getDiceValue(): int
+ printDiveValue (): void

Typecasting : int IntValue = (int) DoubleValue;

Using Math.round()

Math.round() accepts a double value and converts it into the nearest long value by adding 0.50.5 to the value and truncating its decimal points. The long value can then be converted to an int using typecasting.

```
long Math.round(double DoubleValue);
```



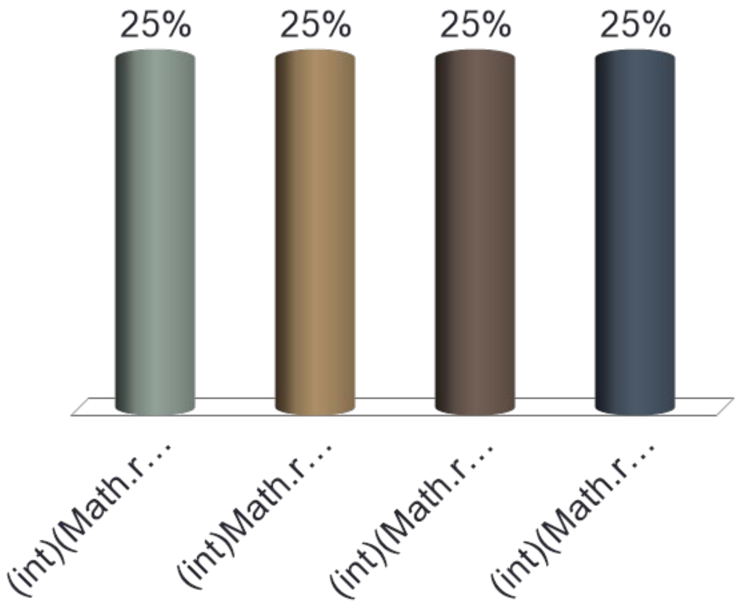
# The java.lang.Math.random() method returns a pseudorandom double type number greater than or equal to 0.0 and less than 1.0. Which of the following code generate a random integer between 1 and 6?

A. (int)(Math.random()*6)

B. (int)Math.random() +6

✓ C. (int)(Math.random()*6) + 1

D. (int)(Math.random()*5) + 1



java.util.Random.nextInt() :

Returns a pseudorandom, uniformly distributed int value between 0 (inclusive) and the specified value (exclusive), drawn from this random number generator's sequence.

Which of the following code generate a random integer between 1 and 6? Assume we already have :

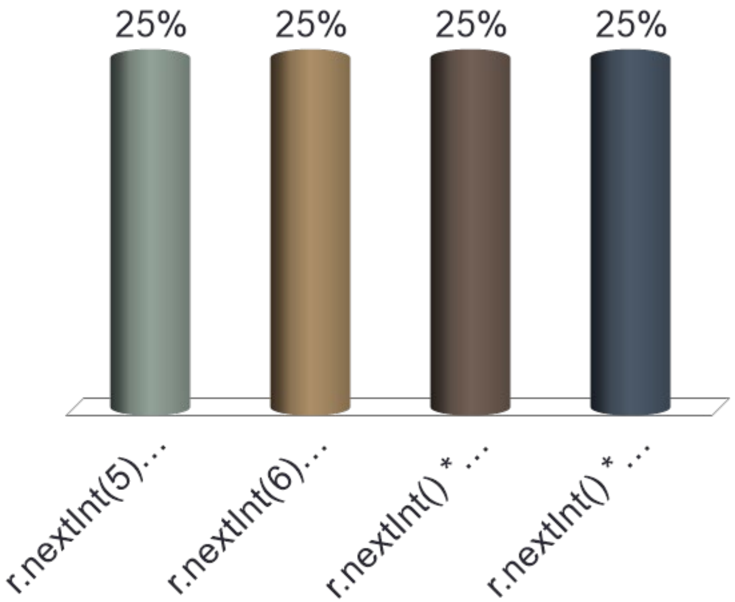
```
import java.util.Random
Random r = new Random();
```

A. r.nextInt(5) + 1 ;

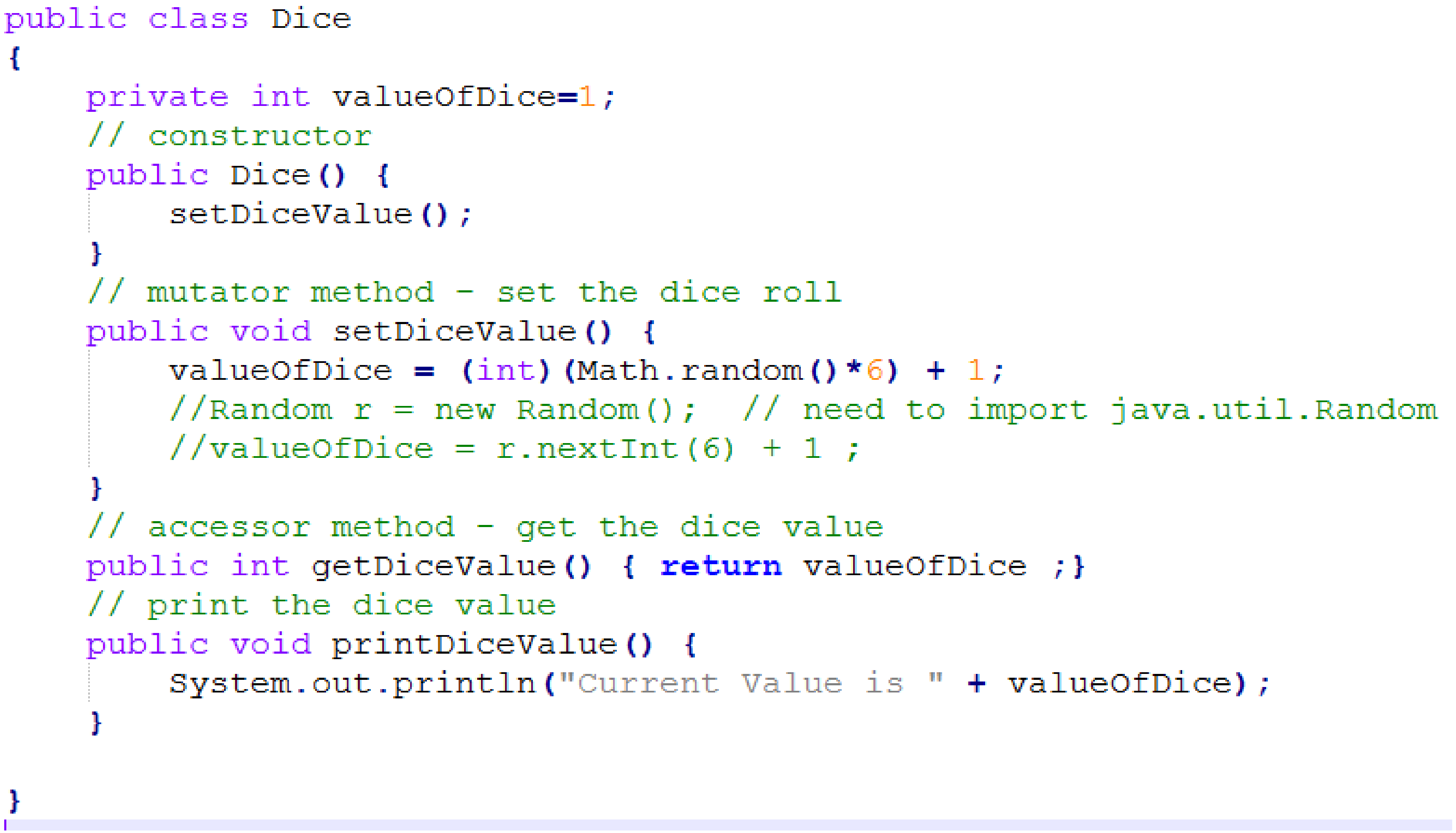
✓ B. r.nextInt(6) + 1 ;

C. r.nextInt() *6 + 1 ;

D. r.nextInt() *5 + 1 ;



# A2



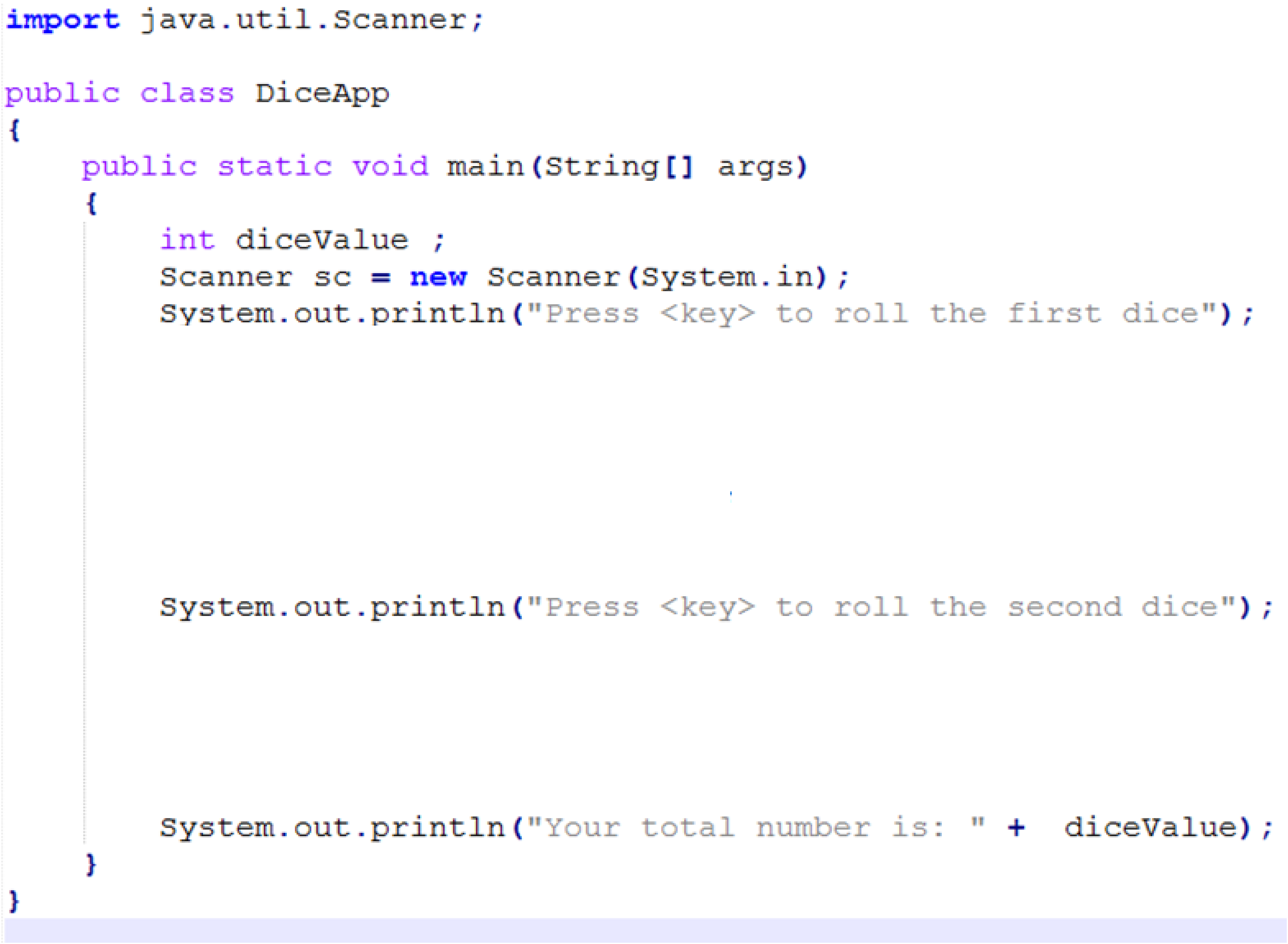
```
public class Dice
{
    private int valueOfDice=1;
    // constructor
    public Dice() {
        setDiceValue();
    }
    // mutator method - set the dice roll
    public void setDiceValue() {
        valueOfDice = (int)(Math.random()*6) + 1;
        //Random r = new Random();  // need to import java.util.Random
        //valueOfDice = r.nextInt(6) + 1 ;
    }
    // accessor method - get the dice value
    public int getDiceValue() { return valueOfDice ;}
    // print the dice value
    public void printDiceValue() {
        System.out.println("Current Value is " + valueOfDice);
    }

}
```

# Q2

- Write an application class `DiceApp` to test the class `Dice`.
- The class `DiceApp` interacts with the user to generate the numbers randomly from rolling a pair of dices.
- The generated numbers from the pair of dices and the total is then displayed on the screen.

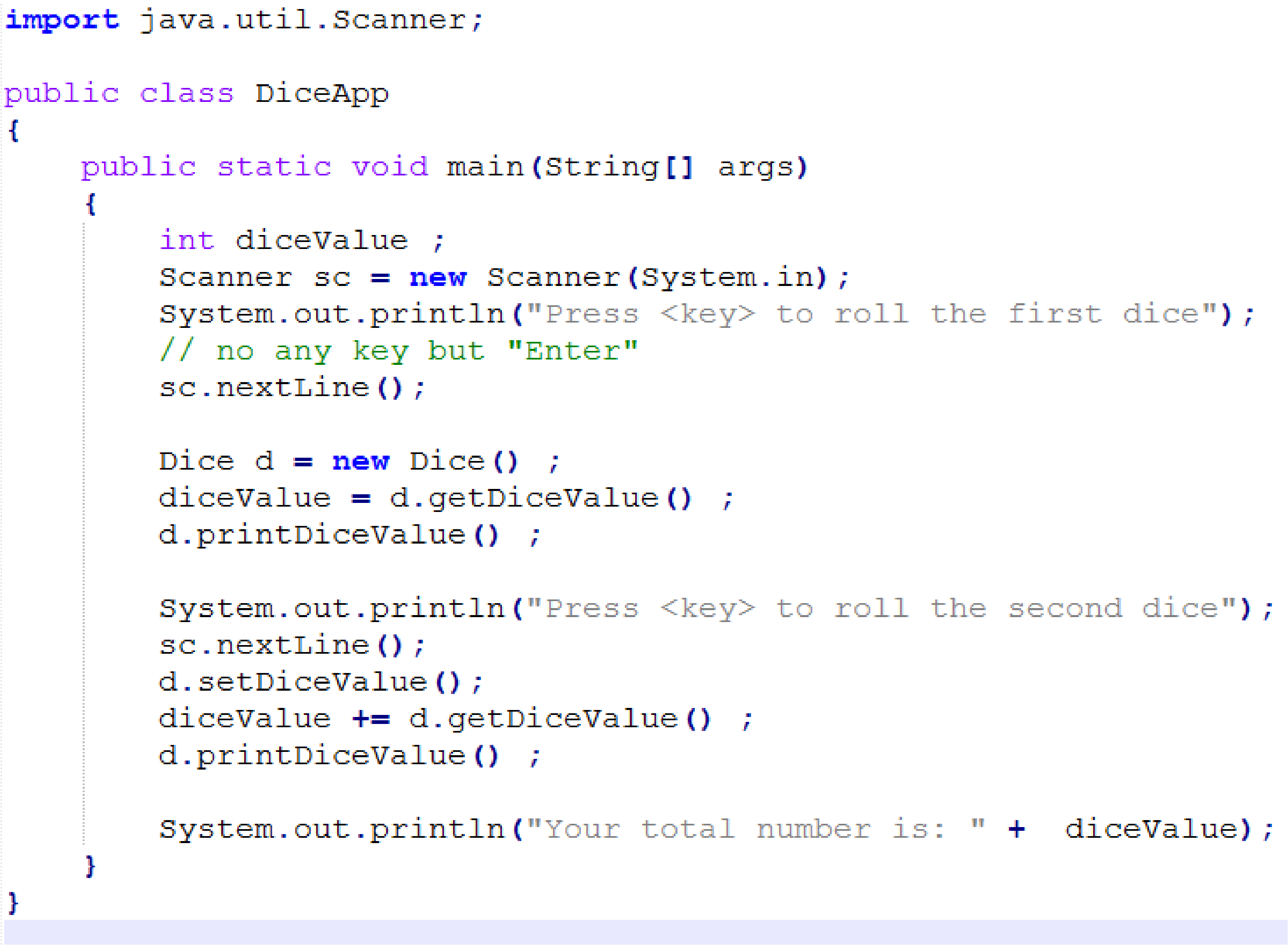
# Q2

```
Press <key> to roll the first dice
1
Current Value is 3
Press <key> to roll second dice
2
Current Value is 3
Your total number is: 6
```



```java
import java.util.Scanner;

public class DiceApp
{
    public static void main(String[] args)
    {
        int diceValue ;
        Scanner sc = new Scanner(System.in);
        System.out.println("Press <key> to roll the first dice");







        System.out.println("Press <key> to roll the second dice");





        System.out.println("Your total number is: " +  diceValue);
    }
}
```

# A2



```
import java.util.Scanner;

public class DiceApp
{
    public static void main(String[] args)
    {
        int diceValue ;
        Scanner sc = new Scanner(System.in);
        System.out.println("Press <key> to roll the first dice");
        // no any key but "Enter"
        sc.nextLine();

        Dice d = new Dice() ;
        diceValue = d.getDiceValue() ;
        d.printDiceValue() ;

        System.out.println("Press <key> to roll the second dice");
        sc.nextLine();
        d.setDiceValue();
        diceValue += d.getDiceValue() ;
        d.printDiceValue() ;

        System.out.println("Your total number is: " +  diceValue);
    }
}
```