King Saud University College of Computer and Information Sciences Computer Science Department	
CSC 111 Introduction to Programming with Java	First Semester 1440-1441

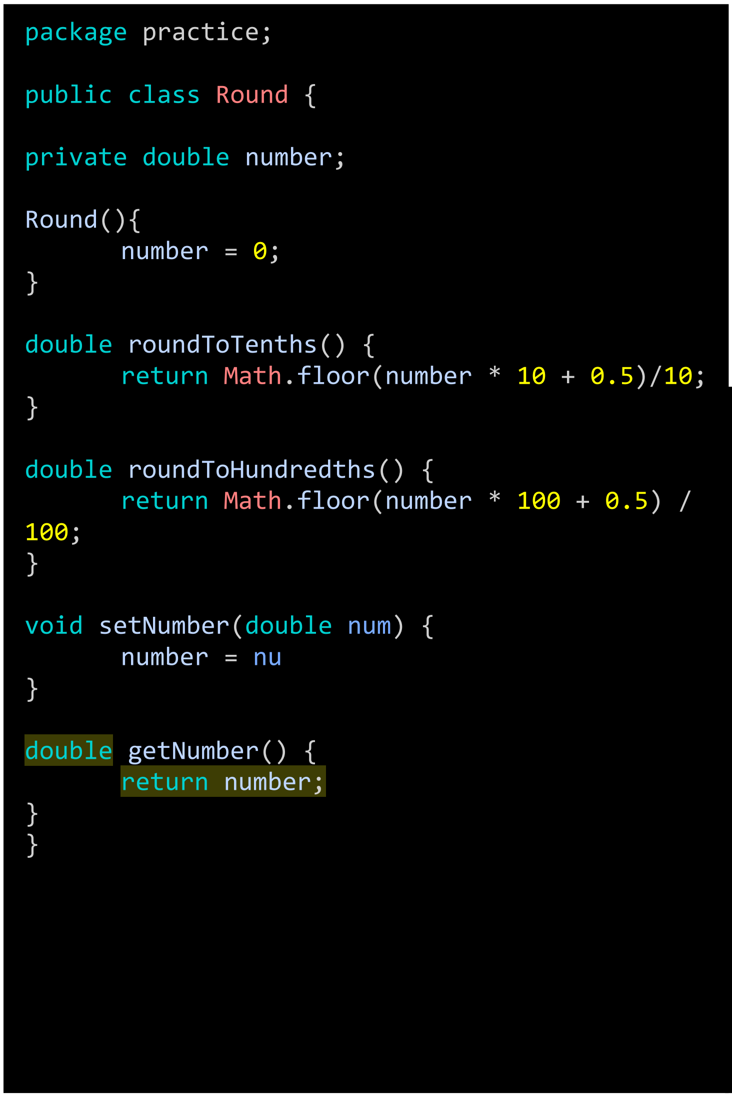
OOP-Sheet

Q1. Write the class **Round**, which has a private attribute called *number*, a default constructor to set the number to 0, and two methods for rounding the number using *Math.floor()*, which is used to round a number to a specific decimal place. In addition the class **Round** has set() and get() methods to handle the private attribute.

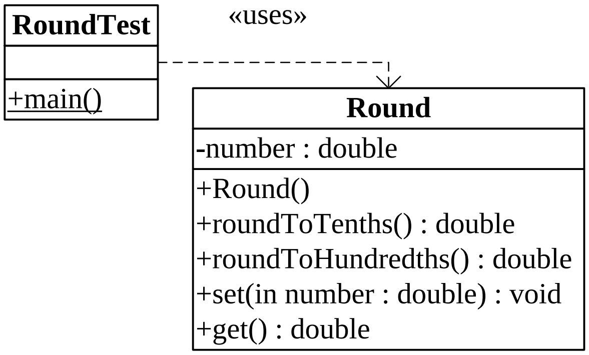
Write another class called **RoundTest** with a **main()** method to create an object of the **Round** class, read the desired number, and invoke all the methods in the class **Round**.

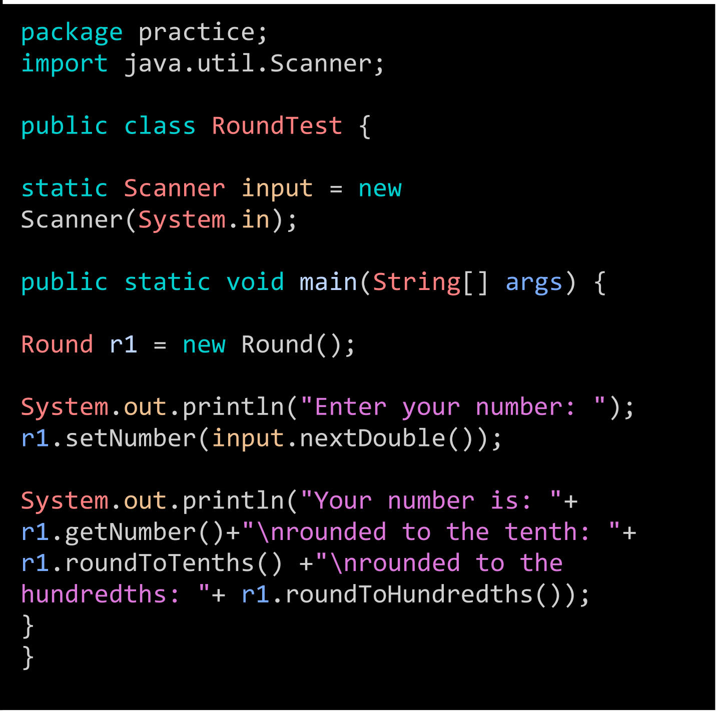
UML classes are represented by the diagram shown below.

Hint: To round to the tenths position use *Math.floor(x * 10 + 0.5) / 10*, and to round to the hundredths position use *Math.floor(x * 100 + 0.5) / 100*.



```
package practice;

public class Round {

private double number;

Round(){
       number = 0;
}

double roundToTenths() {
       return Math.floor(number * 10 + 0.5)/10;
}

double roundToHundredths() {
       return Math.floor(number * 100 + 0.5) /
100;
}

void setNumber(double num) {
       number = nu
}

double getNumber() {
       return number;
}
}
```


RoundTest
+main()
«uses»
Round
-number : double
+Round()
+roundToTenths() : double
+roundToHundredths() : double
+set(in number : double) : void
+get() : double

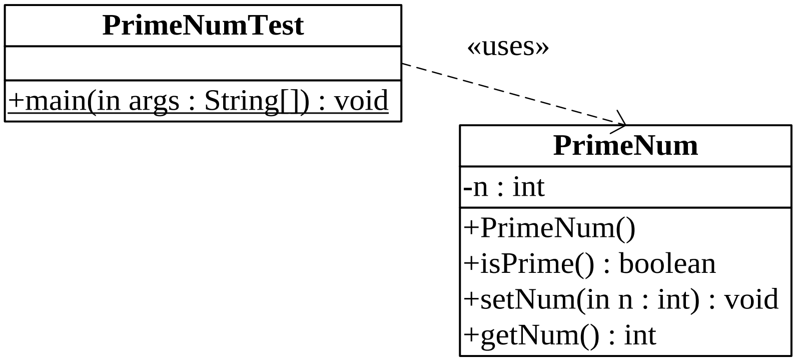


```
package practice;
import java.util.Scanner;

public class RoundTest {

static Scanner input = new
Scanner(System.in);

public static void main(String[] args) {

Round r1 = new Round();

System.out.println("Enter your number: ");
r1.setNumber(input.nextDouble());

System.out.println("Your number is: "+
r1.getNumber()+"\nrounded to the tenth: "+
r1.roundToTenths() +"\nrounded to the
hundredths: "+ r1.roundToHundredths());
}
}
```

Q2.Write a class called **PrimeNum** which has a private attribute called *n*. The **PrimeNum** class has a default constructor that sets the private attribute to 2, and set() and get() methods for the private attribute. In addition, the **PrimeNum** class has a method *isPrime()* that determines whether the number *n* is a prime number or not.

Write a another class called **PrimeNumTest** with a **main()** method that uses the class **PrimeNum** to determine and display all the prime numbers less than a specific integer number entered by a user. UML classes are represented by the diagram shown below.

Hint: A prime number is a natural number that has exactly two natural number divisors which are 1 and itself.



```java
package practice;

public class PrimeNum {
private int n;

PrimeNum(){
      n = 2;
}

void setPrimeNumber(int num) {
      n = num;
}

int getPrimeNumber() {
      return n;
}

boolean isPrimeNumber() {
      for(int i = 2; i < n; i++) {
            if(n%i==0) {
                  return false;
            }
      }
      return true;

}

}
```

```java
package practice;

import java.util.Scanner;

public class PrimeNumTest {
static Scanner input = new Scanner(System.in);

public static void main(String[] args) {
PrimeNum prime1 = new PrimeNum();
System.out.println("Enter a number: ");
int num = input.nextInt();
prime1.setPrimeNumber(num);
if(prime1.isPrimeNumber()) {
System.out.println("The number you entered is " +num+
" and the prime numbers less than "+num+" are:");
PrimeNum[] p = new PrimeNum[num];
for(int i = 0; i < num; i++) {
p[i] = new PrimeNum();
p[i].setPrimeNumber(i+2);
}
for(int j = 0; j < num; j++) {
if(p[j].isPrimeNumber() &&p[j].getPrimeNumber()!=num)

System.out.println(p[j].getPrimeNumber());
} }
else
System.out.println("The number you entered is not
prime");
}
}
```