Java Exercise Questions:

Java Basics – First Program:

1. Given the statement:

```
System.out.println("Hello, World!");
```

What is the name of:

a) a method:
b) a class:
c) an argument:

2. What's wrong with each of the following statements?

a) `System.out.println("Hello, World!")`

b) `System.out.print(Hello World);`

c) `system.out.println("Hello, World!");`

3. What is the **`public class ClassName { … }`** for?

4. What is the purpose of the `main()` method in a Java program?

5. What is displayed by the following statements?

a) `System.out.println("Hello!\nMy name is Mr. Bibs!");`

b) `System.out.println("2 plus 3 is " + 2 + 3);`

6. What's the difference between System.out.print() and System.out.println()?

7. What is a class? How do you recognize a class in your Java code?

8. a) What is the purpose of program comments or “internal program documentation”?

b) What are the two basic ways to create program comments in Java?

9. a) What is the complete path and file name of the Java compiler?

b) What is the complete path and file name of the Java interpreter?

Write a Java application that displays the following information about yourself on the console(i.e. not using dialogs)

Your first and last name

Your previous programming experience (how long? what language(s)?)

Your favourite subject in high school (or university, or wherever you were before this)

Your least favourite subject in high school (or university, or wherever you were before this)

A list of interests or things you enjoy doing in your spare time

Anything else interesting about yourself

Programs:

1. Create a program that displays a list of your three favourite foods, one food per line. Example:

```
Sydney's Favourite Foods:
Chicken
Deli Turkey
Scrambled Eggs
```

2. What is the minimum number of statements you could use in the main() method to produce the output from question 1?

3. Reformat the following code segments by hand so that they follow proper standards:

a)

```
public class BadCode { public static void main(String[]
args){ System.out.println("Flowers for Algernon");}}
```

b)

```
public class
BadCode { public static void main(String[]
args){ System.out.print("6/4*2 is ");System.
```

```
out.println(6/4*2);}}
```

4. Create a program called Tree that creates the following output:

```
    *
   ***
  *****
*******
```

OOP: Object Oriented Programming

1. Define each of the following terms:

a. object

b. instance

c. attribute

d. state of an object

e. behaviour of an object

2. Why do we say that a class is like a template or a recipe?

3. a. What is happening in the following statement:

```
Invoice custInvoice = new Invoice("2006/09/13", "123-c");
```

b. What does the "new" operator do?

4. How do we define "behaviours" in a class?

5. For each of the following things below, list some of the attributes or properties you might define if that thing were an object in a program.

a. Invoice

b. TVShow

c. HockeyTeam

1. Find the errors in each of the following code listings:

a. Listing a)

```
1  public class Qu2PartA
2  {
3      public static void main(String[] args)
4      {
5          width = 15
6          area = length * width;
7          System.out.println("The area is " + area);
8      }
9  }
```

b. Listing b)

```
1   public class Qu2PartB
2   {
3       public static void main(String[] args)
4       {
5           int length, width, area;
6           area = length * width;
7           length = 20;
8           width = 15;
9           System.out.println("The area is " + area);
10      }
11  }
```

c. Listing c)

```
1  public class Qu2PartC
2  {
3      public static void main(String[] args)
4      {
5          int length = 20, width = 15, area;
6          length * width = area;
7          System.out.println("The area is " + area);
8      }
9  }
```

2. Write a program that calculates the amount of money to tip a wait person by coding the following tasks:

1. Define a variable for the bill amount and initialize it to 35.10.
2. Define a variable for the tip percentage and initialize it to 15.
3. Define a variable for the tip amount.
4. Calculate the tip as the bill amount multiplied by the tip percentage (remember that 15% = 0.15) and assign and store the result in the tip amount variable.
5. Display the output on the screen as shown below:

Bill Amount: 35.1 Tip%: 15.0
Tip Amount: $5.265

Circle the correct answer.

1. T / F A float value can be implicitly cast into an int variable because they're both 4 bytes.

2. T / F System.out.println("5 + 1 is " + 5 + 1); will print "5 + 1 is 6".

3. T / F The format specifier %d is used to format integer values.

4. A long must be explicitly cast into a ----- variable.
- **A.** double
- **B.** char
- **C.** float
- **D.** All of the above

5. The Scanner class is in the ---- package.
- **A.** java.lang
- **B.** javax.swing
- **C.** java.util
- **D.** java.io

6. Which Scanner method will read in a whole number with no decimals?
- **A.** nextString()
- **B.** nextNumber()
- **C.** nextDouble()
- **D.** nextInt()

Short Answer Questions (answer in the space provided)

7. Write the following expression in proper Java syntax. Use math methods where appropriate.

$$\frac{2c - \sqrt{(a + b^3)}}{b}$$

8. What is the output of the following code segment?

```
int n1 = 3;
char c1 = 'F';
c1++;
System.out.print(c1);
c1 += n1;
System.out.println(c1);
```

9. For each of the statements below, identify if the statement is valid or invalid. If invalid, rewrite the statement correctly (there might be more than one way to write a statement correctly, just pick one).

```
// use these variable values:
       int number = 5;
       double area = 3.28;
```

b) `long lngValue = area;`

c) `int iNum = area;`

a) `double dblNumber = number;`

10. There are three errors in the code below. For each error, describe the error and indicate which of the three error types it is (Execution, Compilation, or Logic)

```
01: import javax.swing.JOptionPane;
02: public class QuizQuestion {
03: public static void main(String[] args) {
04: String strPrice = JOptionPane.showInputDialog(null,
05: "Enter price of wine:", "Price", JOptionPane.INFORMATION_MESSAGE);
06: double price = Double.parseDouble(strPrice);
07: String strNumBottles = JOptionPane.showInputDialog(null,
08: "Enter number of bottles:", "Inventory",
09: JOptionPane.INFORMATION_MESSAGE);
10: int numBottles = Integer.parseDouble(strNumBottles);
11: // calculate cost of all bottles in inventory
12: double totalCost = price + numBottles;
13: System.out.printf("%-10s %5.2f \n" +  "Total Cost:" + totalCost);
14: } }
```

Line #	Description of Error	Type of Error

Bonus The term "bug" or "debug" is attributed to:

- **A.** Bill Gates
- **B.** Ada Lovelace
- **C.** Steve Jobs
- **D.** Grace Hopper

1. Examine the code listing below. What do you think the output should be? Write this down. Then run the program and see if you're correct.

```
1
2       public class Question2
3       {
4             public static void main(String[] args)
5             {
6               int factor = 2;
7               int sum = 10;
8               System.out.println("sum is " + sum);
9               sum *= factor;
10              System.out.println("sum is now " + sum);
11              sum *= factor;
12              System.out.println("sum is now " + sum);
13              sum *= factor;
14              System.out.println("sum is now " + sum);
15          }
        }
```

2. How do you think the output would change if you wrote the program in question 1 like this:

```
1
2       public class Question3
3       {
4           public static void main(String[] args)
5           {
6               int factor = 2;
7               int sum = 10;
8               System.out.println("sum is " + sum);
9               sum *= factor;
10              sum *= factor;
11              sum *= factor;
12              System.out.println("sum is now " + sum);
13              System.out.println("sum is now " + sum);
14              System.out.println("sum is now " + sum);
15          }
        }
```

Logical Operations

1. Given that a = 5, b = 2, c = 4, and d = 5, what is the result of each of the following Java expression?

a. a == 5

b. b * d == c * c

c. d % b * c > 5 || c % b * d < 7

d. d % b * c > 5 && c % b * d < 7

2. Given that: a = 5 b = 2 c = 4 d = 6 e = 3 What is the result of each of the following relational expressions?

1. a > b

2. a != b

3. d % b == c % b

4. a * c != d * b

5. d * b == c * e

6. a * b < a % b * c

7. c % b * a == b % c * a

8. b % c * a != a * b

9. d % b * c > 5 || c % b * d < 7

10. d % b * c > 5 && c % b * d < 7

3. For each of the following statements, assign variable names for the unknowns and rewrite the statements as relational expressions.

1. A customer's age is 65 or more.

2. The temperature is less than 0 degrees.

3. A person's height is over 6 feet.

4. The current month is 12 (December).

5. The user enters the name "Fred".

6. The user enters the name "Fred" or "Wilma".

7. The person's age is 65 or more and their sub total is more than $100.

8. The current day is the 7th of the 4th month.

9. A person is older than 55 or has been at the company for more than 25 years.

10. A width of a wall is less than 4 metres but more than 3 metres.

11. An employee's department number is less than 500 but greater than 1, and they've been at the company more than 25 years.

4. Show the output of the following program:

```
public class Test {
public static void
main(String[] args) {
char x = 'a';
char y = 'c';
System.out.println(++y);
System.out.println(y++);
System.out.println(x > y);
System.out.println(x - y);
}
}
```

1. Copy the code below into a main() method of a new program, and fill in the missing code by following the instructions in the comments:

```
1
2       public class ScannerExercise {
3
4           public static void main(String[] args) {
5
6               // construct a scanner to get keyboard input
7
8
9               // ask the user for a decimal number
10              // (add the stmt to retrieve the value and store in an
11              // appropriate variable)
12              System.out.print("Enter a decimal number: ");
13
14              // calculate the number times itself (the square)
15              // and store in an appropriate variable (which needs
16              // to be declared - see last statement below where
17              // the variable is being used)
18
19              // user wants to see the result, this is finished so
20              // nothing to do here unless you used different variable name)
21              System.out.println("Your number squared is "
22                  + square);
23          }
24      }
25
```

1. Use the Math class documentation as a reference. What Math class methods would you use to perform the following tasks:

a. find the square root of 13
b. find the minimum value of the two numbers stored in the variables dblNum1 and dblNum2
c. find the ceiling of -123.45
d. find the floor of -123.45
e. find the absolute value of -123.45

2. Write a single statement to perform each of the following calculations and store each result in a variable of the appropriate type:

a. The square root of x - y
b. The absolute value of $a^2 - b^2$
c. The area of a circle (pi multiplied by radius-squared)

3. Write each of the following expressions as a single Java statement:

1. $c = \sqrt{a^2 + b^2}$ c equals the root of a squared plus b squared
(a squared plus b squared is all under the square root symbol)
2. $p = \sqrt{|m - n|}$ p equals the square root of the absolute value of the expression
m minus ne
3. $sum = \frac{a(r^n - 1)}{(r - 1)}$ sum equals the result of a division expression: the numerator
is the variable a times the result of the expression r to the power
of n minus 1, and the denominator is the expression r minus 1

1. Write a program that finds the ASCII/Unicode code for each of the following character values:

a. '7'
b. '1'
c. 'a'
d. 'A'
e. 'z'
f. 'Z'
g. '*'

2. Open the following chart in a new browser window/tab: Simple ASCII Table.

a. What is the decimal value for the character 'A'?
b. What is the decimal value for the character 'a'?
c. What do you think would each of the following statements would evaluate to?
   i. (char)('m' - 5)
   ii. (char)('K' + 6)
   iii. (char)('y' - 'V')
   iv. (char)('K' + '*' - 1)
d. What character has a decimal value of 0?

Exercises

1. How would you use casting to solve the problem in the TestData example in the first section?

2. Using the declaration/initialization statements below, determine the result of each of the following explicit casts.

```
double dNum1 = 5.5;
double dNum2 = 10.2;
double dNum3 = 1.1;
```

a. (int)dNum1
b. (int)dNum2

c. (int)dNum3
d. (int)(dNum1 + dNum3)
e. (int)dNum1 + dNum3
f. (double)((int)dNum1) + dNum3
g. (double)((int)(dNum1 + dNum3))

Exercise

Copy the following program. Compile it, and run it. Test the program with each of the input values below, and for each test, describe what happens and why.

1. 5
2. 5.0
3. five

```
1 public class Conversions {
2
3     public static void main(String[] args) {
4
5         Scanner in = new Scanner(System.in);
6         System.out.print("Enter a value: ");
7         String strValue = in.next();
8         int intNum = Integer.parseInt(strValue);
9         double dblNum = Double.parseDouble(strValue);
10        System.out.println(intNum + ", " + dblNum);
11    }
}
```

1. a) What is a multi-sided if-statement used for?

b) How many conditions does a multi-sided if-statement have?

2. a) How is a switch statement similar to a multi-sided if-statement?

b) Which data types can a switch statement evaluate?

c) What is the purpose of the break; statement in a switch statement?

3. a) What is the output of the following code:

```
public class Question3 {
  final static int NUM_HRS = 44;
  public static void main(String[] args) {
    double rate;
    int dept = 5;
    switch (dept) {
      case 1: case 2: case 3:
        rate = 10.5;
        break;
      case 4: case 5:
        rate = 19.95;
        break;
      case 6: case 7: case 8: case 9:
        rate = 14.55;
        break;
      default:
        rate = 0;
    }
    double total = rate * NUM_HRS;
    System.out.printf("Total Pay: $%.2f%n", total);
  }
}
```

3. b) Rewrite the code in 3.a) as an if/else-if.

4. What is the error in each of the following code segments?

a)

```
double area = 0;
if (length > 0);
{
   area = length * 25.5;
   System.out.printf("Area: %.2f%n", area);
}
```

b)

```
Scanner scan = new Scanner(System.in);
System.out.print("Enter a whole number: ");
int num = scan.nextInt();
if (num % 2 = 0)
    System.out.println("Your number is even.");
else
    System.out.println("Your number is odd.");
```

1) A. Write a loop that prints the numbers from 1 to 10 with their squares, like this:

```
1       1
2       4
3       9
4       16
5       25
6       36
7       49
8       64
9       81
10      100
```

B. How would you make the loop do the same output backwards (from 10 to 1)?

2) Write a while loop that counts from 1 to 20 by 2's on one line:

```
2 4 6 8 10 12 14 16 18 20
```

3) What is the output of each of the following loops?

Example 1:

```
int count = 4;
while (count > 0)
{
  System.out.println(count);
  count--;
}
```

Example 2:

```
int count = 1;
while (count < 5)
{
  System.out.println(count);
  count++;
}
```

Example 3:

```
int count = 3;
while (count <= 1)
{
  System.out.println(count);
  count++;
}
```

Example 4:

```
int count = 3;
while (count >= 1)
{
  System.out.println(count);
  count--;
}
System.out.println(count);
```

Example 5:

```
int count = 9;
while (count <= 10 && count > 4)
{
  System.out.println(count);
  count--;
}
```

Example 6:

```
int x = 1, y = 3;
while (x < 5 || y > 0)
{
  System.out.println(x++ +
        ", " + --y);
}
System.out.print("x: " + x);
System.out.println(" y:" + y);
```

4) a. Write a program that requests a final grade. Use a do-loop to request the grade continuously as long as the grade entered is invalid. A grade is invalid if it is less than 0 or greater than 100. After a valid grade is entered, display it on the screen.
4) b. Modify the above program so that the user can cancel by entering the value 999.
4) c. Modify the program in 1. a. so that the user has only 5 tries to enter the grade. After 5 tries, the program terminates.

5) a. For this program, use either a while-loop or a do-loop, whichever you think is most efficient. Write a program that records scientific test results (they'll have decimal values) from the user. As each test result is entered, add it to a total. After all the test results have been entered, display the total. You don't know how many test results they'll enter, so after each result is entered, prompt the user with a question such as "Would you like to enter another test result? (Y/N)". The user can answer "Yes" or "No" too this question, or even just "Y" or "N". To capture this, we would use the Scanner's next() method to grab only the first word of the user input:

```
String answer = in.next();
```

However, we can make this program more efficient if we use a character value instead of a string, and just compare the first letter of the user's input (Y or N). There is a method in the String class called charAt(index). You give charAt() a string index (the position number, where the first character is position 0, the second position 1, etc) of the character you'd like to have and it will return that character at the specified index as a char value. For example:

```
"hello".charAt(0) // returns 'h'
"hello".charAt(1) // returns 'e'
"hello".charAt(2) // returns 'l'
"hello".charAt(4) // returns 'o'
"hello".charAt(5) // would give an error because there is no index 5
```

So since we only what the first character that the user types, we can use:

```
char keepGoing = 'Y';  // initialize to yes
...  // code code code
keepGoing = in.next().charAt(0);
```

Next, we want to see if the user says 'Y' to our prompt, meaning they'd like to enter another test result. So our loop condition could be something like keepGoing == 'y' || keepGoing == 'Y' because we don't know if the user will type an upper-case or a lower-case answer. If you prefer, you can use one of the String class's methods toUpperCase() or toLowerCase() to convert the user's answer to upper- or lower-case, and then compare. For example, I will use upper-case comparisons:

```
keepGoing = in.next().toUpperCase().charAt(0);
```

Then my loop condition would be keepGoing == 'Y'.
Add this functionality to your program, so your user can enter as many test results as they'd like.

1. Using loop statement write a program that prompts the user to enter 5 integer values:

i. Find and display the Largest and Smallest number

ii. Display whether the number is Even or Odd

iii. Display whether the number is negative, positive or zero

iv. Calculate the Sum and Average of the Even numbers

2. Write a program that randomly generates an integer between 0 and 100, inclusive.

The program prompts the user to enter a number continuously until the number matches the randomly generated number. For each user input, the program tells the user whether the input is too low or too high, so the user can choose the next input intelligently.

- Object Oriented Programming 1

Exercise

Part A.

A.1 Write a statement to display 3.5 ^ 9.2.

A.2 Write an if-else statement for y = |x – 1|. Assume that x and y are already declared.

A.3 Assume that x is an int variable. Write a conditional expression to fill in the following blank for the statement that displays "x is even" if x is even, or "x is odd" if x is odd.

```
System.out.println("x is " + _____________________________________);
```

Part B. Show the output of the following code: (write the output next to each println statement if the println statement is executed in the program).

```
public class Test {
  public static void main(String[] args) {

    System.out.println((int)(Math.random()));

    System.out.println(Math.pow(2, 3));

    System.out.println(34 % 7);

    System.out.println(3 + 4 * 2 > 2 * 9);

    int number = 4;

    if (number % 3 == 0)
      System.out.println(3 * number);

    System.out.println(4 * number);

    int x = 943;
    System.out.println(x / 100);

    System.out.println(x % 100);

    System.out.println(x + " is " + ((x % 2 == 0) ? "even" : "odd"));

    int y = -1;
```

```
    y++;
    System.out.println(y);
  }
}
```

Part C:

1. Write a program that prompts the user to enter the exchange rate from currency US dollars to Chinese RMB. Prompt the user to enter 0 to convert from US dollars to Chinese RMB and 1 vice versa. Prompt the user to enter the amount in US dollars or Chinese RMB to convert it to Chinese RMB or US dollars, respectively. Here are the sample runs:

<Output>
Enter the exchange rate from dollars to RMB: 6.81
Enter 0 to convert dollars to RMB and 1 vice versa: 0
Enter the dollar amount: 100
$100.0 is 681.0 Yuan
<End Output>

<Output>
Enter the exchange rate from dollars to RMB: 6.81
Enter 0 to convert dollars to RMB and 1 vice versa: 1
Enter the RMB amount: 10000
10000.0 Yuan is $1468.43
<End Output>

<Output>
Enter the exchange rate from dollars to RMB: 6.81
Enter 0 to convert dollars to RMB and 1 vice versa: 5
Incorrect input
<End Output>

2. Write a program that prompts the user to enter an integer. If the number is a multiple of 5, print HiFive. If the number is divisible by 2 or 3, print Georgia. Here are the sample runs:

<Output>
```
Enter an integer: 6
Georgia
```
<End Output>

<Output>
```
Enter an integer: 15
HiFive Georgia
```
<End Output>

<Output>
```
Enter an integer: 25
HiFive
```
<End Output>

<Output>
```
Enter an integer: 1
```
<End Output>

Part D: Multiple Choice Questions:

1. The expression (int)(76.0252175 * 100) / 100 evaluates to _____.

a. 76
b. 76.0252175
c. 76.03
d. 76.02

#

2. Assume x is 0. What is the output of the following statement?

```
if (x > 0)
    System.out.print("x is greater than 0");
else if (x < 0)
    System.out.print("x is less than 0");
else
    System.out.print("x equals 0");
```

a. x is less than 0
b. x is greater than 0

c. x equals 0
d. None

#
3. Analyze the following code:

Code 1:

```
boolean even;

if (number % 2 == 0)
   even = true;
else
   even = false;
```

Code 2:

```
boolean even = (number % 2 == 0);
```

a. Code 2 has syntax errors.
b. Code 1 has syntax errors.
c. Both Code 1 and Code 2 have syntax errors.
d. Both Code 1 and Code 2 are correct, but Code 2 is better.

#
4. What is x after evaluating

```
x = (2 > 3) ? 2 : 3;
```

a. 5
b. 2
c. 3
d. 4

#
5. Analyze the following code.

```
int x = 0;
if (x > 0);
{
   System.out.println("x");
}
```

a. The value of variable x is always printed.
b. The symbol x is always printed twice.
c. The symbol x is always printed.

d. Nothing is printed because x > 0 is false.

#

6. To declare a constant MAX_LENGTH inside a method with value 99.98, you write

a. final double MAX_LENGTH = 99.98;
b. double MAX_LENGTH = 99.98;
c. final MAX_LENGTH = 99.98;
d. final float MAX_LENGTH = 99.98;

#

7. Which of the following is a constant, according to Java naming conventions?

a. read
b. MAX_VALUE
c. ReadInt
d. Test

#

```
8.  Which of the following code displays the area of a circle if the radius is
positive.

a.  if (radius <= 0) System.out.println(radius * radius * 3.14159);
b.  if (radius != 0) System.out.println(radius * radius * 3.14159);
c.  if (radius >= 0) System.out.println(radius * radius * 3.14159);
d.  if (radius > 0) System.out.println(radius * radius * 3.14159);
```

Exercises

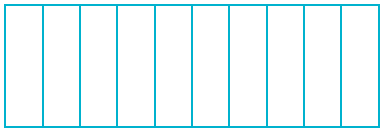
1. For each of the following, declare an array with an appropriate name and type and allocate an appropriate amount of storage.

a. a list of grades for 100 courses
b. A list of 10 names
c. a list of 50 temperatures
d. a list birth years for 25 club members
e. a list of 200 product IDs (product IDs can include digits, letters, and the dash (-))
f. a list that keeps track of the numbers of students in 5 different classes (e.g. class 1 has x students, class 2 has y students, etc)

2. Fill in the elements for the **values[]** array (it has 10 elements) as the following code executes:

```
int counter = 1;
values[0] = 10;
values[counter] = counter;
counter++;
values[5] = counter;
values[9] = values[5] + counter;
values[counter] = values[9] - values[1];
values[9] += ++counter;
```

Array: values[10]



3. Write the code to display the **values[]** array (from the previous exercise) backwards.

4. a. Write a program that uses a char[] array to store the characters in a sentence. Ask the user for a sentence, and then store that string into the char[] array.

b. Modify the above program to do a search/replace for one letter in the char[] array. Ask the user what letter they'd like to replace, and then the letter they'd like to replace it with. After doing the search/replace, display the sentence.

Exercises

1. Record 10 integer values from the user and store them in an array. After recording the 10 values, calculate and display:

- The average
- The highest value
- The lowest value

2. Find out how often the numbers from 1 to 10 are generated randomly. Declare an array to hold 10 integer elements. Generate 100 random numbers from 1 to 10 inclusive, and use the array to keep track of how many times each number occurs. For example, if the first number generated is 9, add 1 to the 9th element of the array. If the second number generated is 3, add 1 to the 3rd element of the array. If the third number generated is 9, add another 1 to the 9th element of the array. After 100 numbers have been generated, each element of the array will hold the number of times that value was generated. Display the array.

In this example, we're using the array as a **frequency table** to keep track of the frequency of each of the 10 numbers.

3. A small school is keeping track of the number of cans recycled for it's grade 1, 2, and 3 classes. Each time students bring in cans, the number of cans are counted and added to the total for that student's class or grade.

Use an array to keep track of the totals for each of the three grades (therefore, the array should have 3 elements). The user will be repeatedly prompted to enter the grade number, and the number of cans brought by a student, until they decide to quit. For each number of cans entered, add that number to the total for that grade (e.g. the proper array element).

After the user is finished entering data, display the totals for the three grades.

4. Ask the user to enter five integer values. Store the values in an array and then determine if the values were entered in ascending order. Display a message indicating whether they are sorted or not.

Order of Precedence

1. In the table below there are 5 sets of algebraic expressions and a corresponding incorrect Java expression. Find the errors in the Java expressions and rewrite them correctly.

	Algebraic Expression	Incorrect Java Expression	Correct Java Expression
1.	(2)(3) + (4)(5)	`(2)(3)+(4)(5)`	
2.	$\frac{6 + 18}{2}$	**6 + 18 / 2**	
3.	$\frac{4.5}{12.2 - 3.1}$	**4.5 / 12.2 - 3.1**	
4.	4.6(3.0 + 14.9)	**4.6(3.0 + 14.9)**	
5.	(12.1 + 18.9)(15.3 - 3.8)	**(12.1 + 18.9)(15.3 - 3.8)**	

2. Evaluate each of the following integer expressions:

a. **20 - 5 / 2 + 3** = =
b. **20 - 5 / (2 + 3)** = =
c. **10 * (1 + 7 * 3)** = =
d. **15 % 3** = =
e. **10 + 5 % 2** = =
f. **10 * 5 % 2** = =

3. Evaluate each of the following expressions:

a. **6.0 / 4.0** = =
b. **20.0 - 5.0 / 2.0 + 3.0** = =

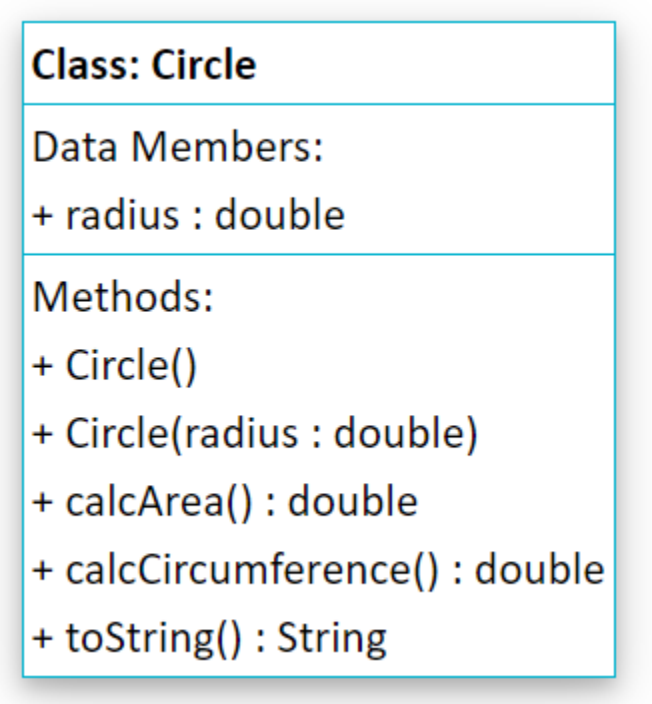
c. **20.0 - 5.0 / (2.0 + 3.0) =** **=**
d. **(20.0 - 5.0) / (2.0 + 3.0) =** **=**

4. Evaluate each of the following expressions:
a. **10.0 + 15 / 2 + 4.3** **=** **=**
b. **10.0 + 15.0 / 2 + 4.3** **=** **=**
c. **3 * 6.0 / 4 + 6** **=** **=**
d. **20.0 - 5 / 2 + 3.0** **=** **=**
e. **3.0 * 4 % 6 + 6** **=** **=**

Exercises

Create each of the following classes based on the class diagram below (in UML, something that starts with a + is to be declared as public; if it starts with a - (minus), it is to be declared as private. We haven't talked about the private modifier yet, so for now we'll declare everything as public. Note that this will change for certain members as we learn more about OOP):

1.

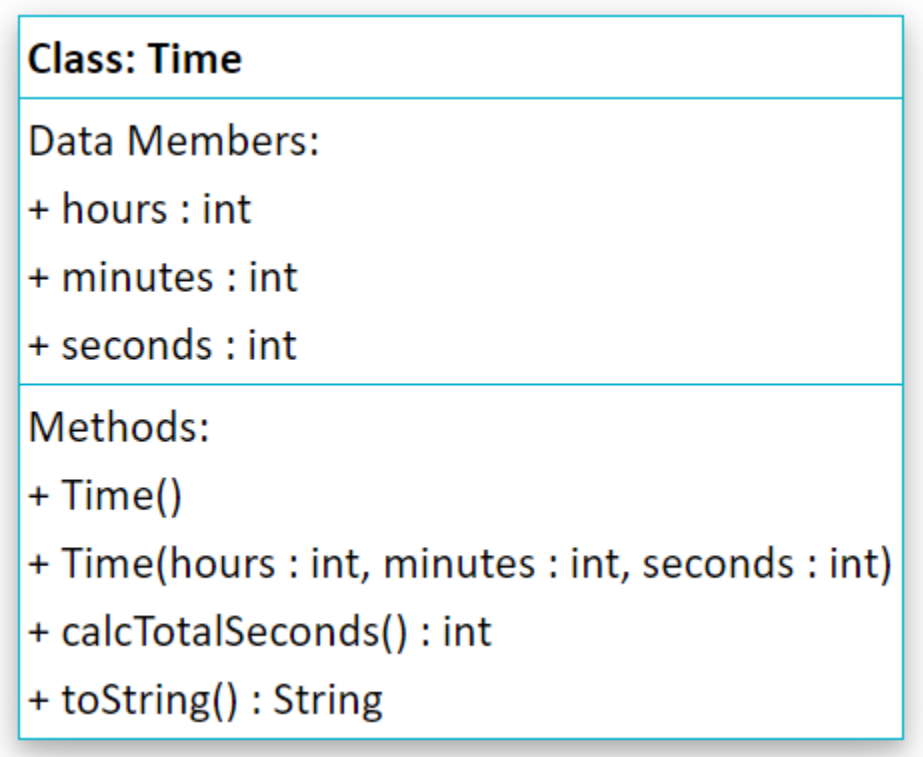


The calcArea() and calcCircumference() methods calculate and return the area and circumference of the circle (Use Google if you need to find the formulas). The default constructor sets radius to a default value of 1. The toString() method returns a string representation of the circle in the form:

```
Circle: radius=x.x
```

(where x.x is the radius value, formatted to 1 decimal place)

2.

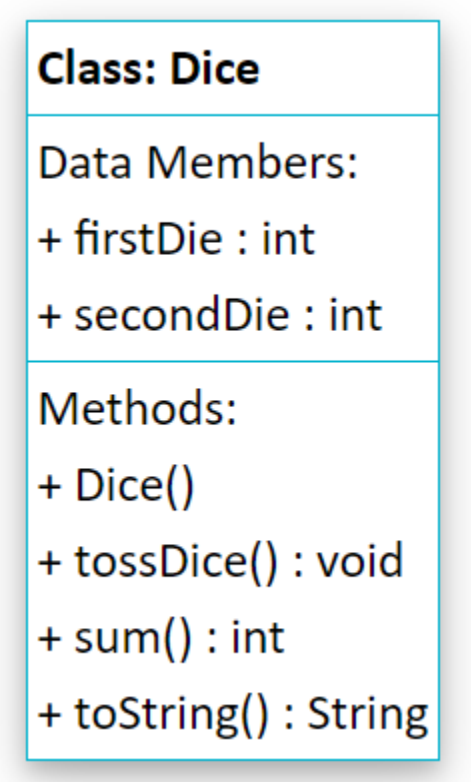


This class models a specific time stamp on a clock. The calcTotalSeconds() method calculates and returns the total number of seconds from midnight to the time represented by the time object (use the normal calculation for total number of seconds: hours * 3600 + minutes * 60 + seconds). The default constructor sets the time to midnight by setting each attribute to 0. The toString() method returns a string representation of the time object in the form:

`hh:mm:ss`

(where hh, mm, and ss are the values of hours, minutes and seconds in the time object, formatted to show exactly 2 digits)

3.



The tossDice() method generates to random integers from 1 to 6 and places one in each instance variable. The sum method returns the sum of the two dice values. The default constructor tosses the dice so that they are initialized with a random set of values. The toString() method returns a string representation of the pair of die in the form:

```
x, y
```

(where x and y are the two instance variable values)

Test the dice class in a Main class: construct the dice and allow the user to play a dice game that is a variation on the classic dice game called "Chicago": A player rolls a pair of dice 11 times. In those 11 rolls, they must attempt to roll each value from 2 to 12. At the end of the 11 rolls, total up the successful rolls to get their score.

In each round, display the dice roll and the sum of that roll in parentheses. For example, if the user rolled a 4 and a 3, then the value 4, 3 (7) is displayed.

After all rounds are completed, display a list of results: List each value from 2 to 12 and whether or not the user managed to roll that value during their 11 rounds.

After displaying the results of the 11 rounds, display the total points earned.

The sample program interaction and output below can better demonstrate how the game works:

```
Starting the game!
Round 2...
6, 3 (9)
Round 3...
1, 5 (6)
Round 4...
4, 4 (8)
Round 5...
6, 1 (7)
Round 6...
3, 1 (4)
Round 7...
1, 1 (2)
Round 8...
4, 3 (7)
Round 9...
4, 4 (8)
Round 10...
3, 4 (7)
Round 11...
5, 5 (10)
Round 12...
5, 2 (7)

Your results:
 2: yes
 3: no
 4: yes
 5: no
 6: yes
 7: yes
 8: yes
```

```
 9: yes
10: yes
11: no
12: no
Your total score: 46
```

Hint: use a boolean array to keep track of which values from 2 to 12 the player has already rolled. Each round, check to see if the array element corresponding to that round number is set to false: if so, then that number hasn't been rolled yet, so you can count it.

PROG10082

1. What is a method?

2. What are some of the reasons why we write methods?

3. What does the following method header say about how this method works?

```
public static double calcGrossPay(int scale, double hours) { … }
```

4. a) What is an argument?

b) What is a parameter variable?

c) What is meant by “return value”?

5. Trace the program below. What is the output? Check your answer by running the program.

```
public class Stuff {
	public static void main(String[] args) {
		int number = 5;
		doStuff();
		System.out.println("Main Number: " + number);
	}

	public static void doStuff() {
		int number = 1;
		System.out.print("New Number: " + number);
		number *= 2;
		System.out.println(", " + number);
	}
}
```

6. In question 5, which variables are local? To which methods are they visible?

Exercises

1. Write a program that asks the user for a sentence. Display the sentence backwards, letter by letter. Example:

```
Enter a sentence:
Mary had a little lamb.
Your sentence backwards:
.bmal elttil a dah yraM
```

2. Write a program that asks the user for a sentence. Then ask the user which character they'd like to replace, and which character to replace the first character with. Create a new string with all instances of the first character replaced with the second character, and display the new sentence on the screen. Example:

```
Enter a sentence:
Mary had a little lamb.
Enter a character to replace:  b
Enter a character to replace it with: p
Your new sentence:
Mary had a little lamp.
```

3. Write a program that asks the user to enter two strings representing an amount of time in the form hh:mm:ss. Then write the code that calculates the total number of seconds in that amount of time. Finally, calculate and display difference in seconds between the two times. For example:

```
Enter an amout of time in the form hh:mm:ss
2:13:55
Enter another amount of time in the form hh:mm:ss
14:03:12

There is a difference of 42,557 seconds between 2:13:55
and 14:03:12
```

Write a method called getTotalSeconds() that accepts a time string in the format hh:mm:ss. This method should extract the number of hours, minutes, and seconds from the string and then use these values to calculate the total number of seconds. The result should be returned.

4. A program that asks the user for a sentence, and then returns that string with the words backwards. For example:

```
Enter a sentence!
Mary had a little lamb.

Your sentence backwards:
lamb. little a had Mary
```

Write a method called reverse() that accepts a sentence as a string, and then returns that string with the words backwards. Write a main() method that gets the string from the user and then displays the string backwards.

5. An array contains 5 strings. Write a program that creates the array (you can hard-code the array or fill it with user inputs), then displays the index numbers of elements that contain the letter "e" in upper-case or lower-case.

6. (advanced!) Validate a Social Insurance Number.

Canada's social insurance numbers (SIN) are validated using a special checksum algorithm that goes something like this:

1. Separate the first 8 digits from the 9th digit. The 9th digit is a check-digit. Take the remaining 8 digits:
2. Multiply the first digit by 1, the second digit by 2, the third digit by 1, the fourth digit by 2, etc. In other words, Each odd-positioned digit (first, third, fifth, and seventh) are multiplied by 1 and each even-positioned digit (second, fourth, sixth, eighth) are multiplied by 2.
3. If any product above results in a 2-digit number, add those digits together to get a single digit.
4. Take the 8 single-digit numbers from the result in step 2 and 3 and add them all together.
5. Take the value in step 4 and find the next highest number that is evenly divisible by 10. Subtract your step 4 solution from that number.
6. If the solution to step 5 is the 9th check-digit, then your SIN is valid. If it isn't the same as your check-digit, then it wasn't a valid SIN.

Example using a ficticious SIN 046 454 286:

Step 1: Separate the first 8 digits from the check digit:

Value to check: 04645428 Check-Digit: 6

Step 2 & 3: Multiply the odd-positioned digits by 1 and the even-positioned digits by 2. For any result that is more than one digit, add the solution's digits together to get a single digit.

```
0 * 1 = 0
4 * 2 = 8
6 * 1 = 6
4 * 2 = 8
5 * 1 = 5
4 * 2 = 8
2 * 1 = 2
8 * 2 = 16, 1 + 6 = 7
```

Step 4: Add the digits in Step 2/3:

0 + 8 + 6 + 8 + 5 + 8 + 2 + 7 = 44

Step 5: Find the next highest number that is evenly divisible by 10 and subtract:

Next highest number from 44 is 50. 50 - 44 = 6

Step 6: The solution to step 5 is 6, which equals our check-digit of 6, so this is a valid SIN.

Write a program that prompts the user for a SIN and then displays whether or not the SIN is valid or invalid.

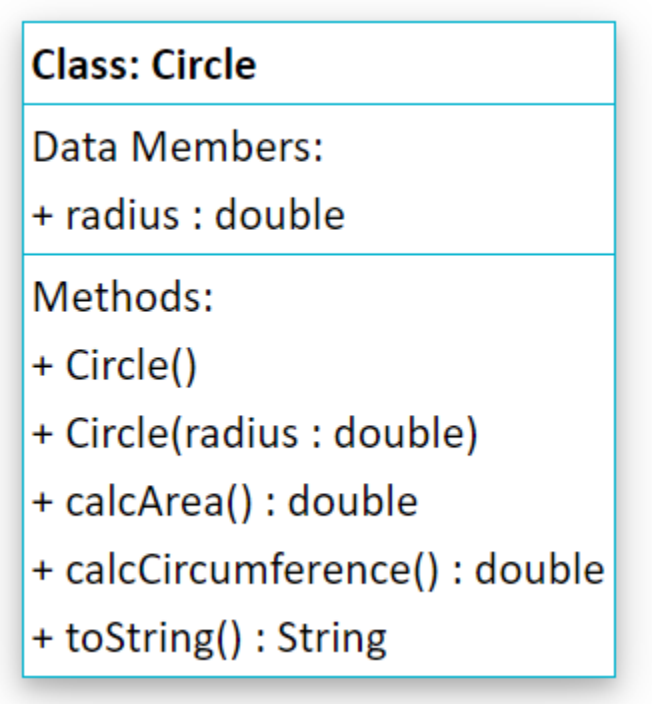
Check your program with these SINs:

410345678
123456030
193456787
123456789

Exercises

Create each of the following classes based on the class diagram below (in UML, something that starts with a + is to be declared as public; if it starts with a - (minus), it is to be declared as private. We haven't talked about the private modifier yet, so for now we'll declare everything as public. Note that this will change for certain members as we learn more about OOP):

1.



The calcArea() and calcCircumference() methods calculate and return the area and circumference of the circle (Use Google if you need to find the formulas). The default constructor sets radius to a default value of 1. The toString() method returns a string representation of the circle in the form:

```
Circle: radius=x.x
```

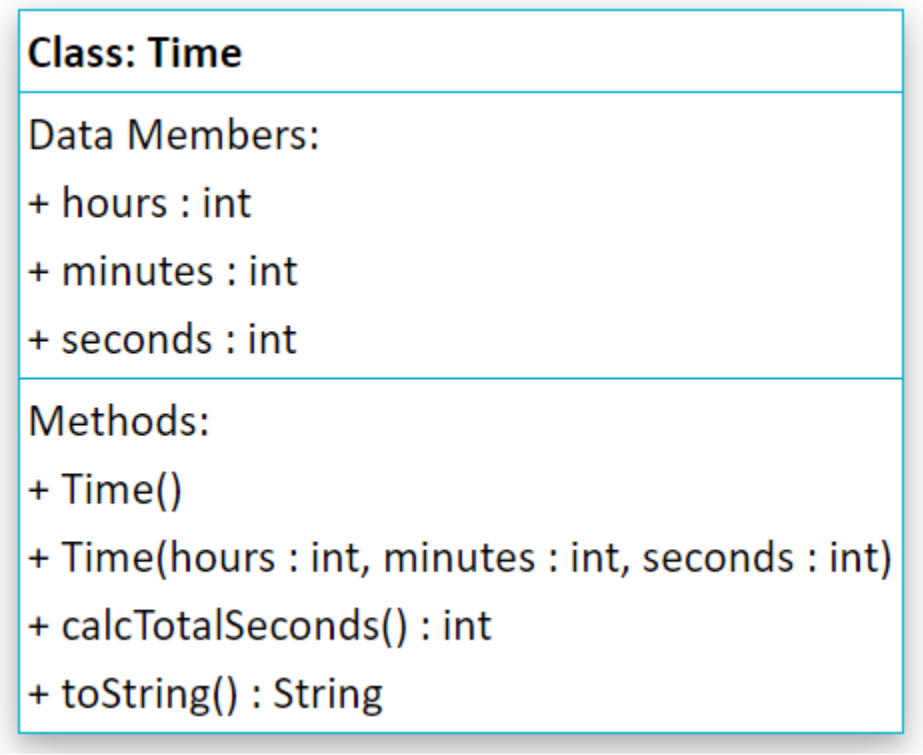
(where x.x is the radius value, formatted to 1 decimal place)

```java
public class Circle {

    // circle's radius, default is 1
    public double radius = 1;

    /*
     * Default constructor, which creates a default
     * Circle object with a radius of 1.
     */
    public Circle() {}

    /*
     * Constructs a Circle object with a programmer-specified
     * value for the circle's radius.
     */
    public Circle(double radius) {

        // set the radius of the circle
        this.radius = radius;
    }

    /*
     * Calculates and returns the area of the circle object.
     */
    public double calcArea() {

        // calculate and return circle area
        return Math.PI * Math.pow(radius, 2);
    }

    /*
     * Calculates and returns the circumference of the circle
     * object.
     */
    public double calcCircumference() {

        // calculate and return circle's circumference
        return 2 * Math.PI * radius;
    }

    /*
     * Returns a String representation of this circle object.
     */
    public String toString() {

        // return a formatted String for this Circle object
        return String.format("Circle: radius=%.1f", radius);
    }
}
```

```
11 public class TestCircle {
12
13     public static void main(String[] args) {
14
15         // test the circle class:
16         // construct a default circle and display it
17         Circle defCircle = new Circle();
18         System.out.println(defCircle);
19
20         // construct a circle with radius 2 and display it
21         // display 2nd circle's area and circumference
22         Circle tire = new Circle(2);
23         System.out.println(tire);
24         System.out.printf("Area: %.2f%n", tire.calcArea());
25         System.out.printf("Circumference: %.2f%n",
26                 tire. calcCircumference());
27     }
28 }
```

the TestCircle class

2.



This class models a specific time stamp on a clock. The calcTotalSeconds() method calculates and returns the total number of seconds from midnight to the time represented by the time object (use the normal calculation for total number of seconds: hours * 3600 + minutes * 60 + seconds). The default constructor sets the time to midnight by setting each attribute to 0. The toString() method returns a string representation of the time object in the form:

```
hh:mm:ss
```

(where hh, mm, and ss are the values of hours, minutes and seconds in the time object, formatted to show exactly 2 digits)

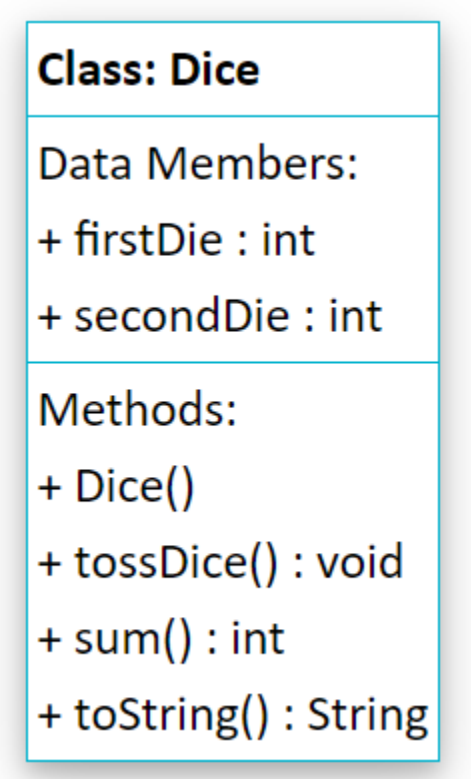
```java
public class Time {

    public int hours;   // number of hours
    public int minutes; // number of minutes
    public int seconds; // number of seconds

    /*
     * Constructs a default time of midnight (00:00:00).
     */
    public Time() {}

    /*
     * Constructs a time object with a specific number
     * of hours, minutes, and seconds.
     */
    public Time(int hours, int minutes, int seconds) {

        // assign the hours, minutes, and seconds
        this.hours = hours;
        this.minutes = minutes;
        this.seconds = seconds;
    }

    /*
     * Calculate the total number of seconds from midnight.
     */
    public int getTotalSeconds() {

        // calculate and return the total seconds
        return hours * 3600 + minutes * 60 + seconds;
    }

    /*
     * Return a String representation of this time object
     * in the form hh:mm:ss
     */
    public String toString() {

        // create and return a formatted time string for
        // this time object
        return String.format("%02d:%02d:%02d", hours,
             minutes, seconds);
    }
}
```

the Time class

```java
public class TestTime {

    public static void main(String[] args) {

        // construct a default time object (midnight)
        Time time1 = new Time();
        System.out.println(time1);

        // construct a time object for 2 minutes and 3
        // seconds past 1 o'clock
        Time time2 = new Time(1, 2, 3);

        // display the time object and the total # of
        // seconds from midnight to this time
        System.out.println(time2);
        System.out.println("Total Seconds: " +
             time2.getTotalSeconds());
    }
}
```

the TestTime class

3.



The tossDice() method generates to random integers from 1 to 6 and places one in each instance variable. The sum method returns the sum of the two dice values. The default constructor tosses the dice so that they are initialized with a random set of values. The toString() method returns a string representation of the pair of die in the form:

```
x, y
```

(where x and y are the two instance variable values)

Test the dice class in a Main class: construct the dice and allow the user to play a dice game that is a variation on the classic dice game called "Chicago": A player rolls a pair of dice 11 times. In those 11 rolls, they must attempt to roll each value from 2 to 12. At the end of the 11 rolls, total up the successful rolls to get their score.

In each round, display the dice roll and the sum of that roll in parentheses. For example, if the user rolled a 4 and a 3, then the value 4, 3 (7) is displayed.

After all rounds are completed, display a list of results: List each value from 2 to 12 and whether or not the user managed to roll that value during their 11 rounds.

After displaying the results of the 11 rounds, display the total points earned.

The sample program interaction and output below can better demonstrate how the game works:

```
Starting the game!
Round 2...
6, 3 (9)
Round 3...
1, 5 (6)
Round 4...
4, 4 (8)
Round 5...
6, 1 (7)
Round 6...
3, 1 (4)
Round 7...
1, 1 (2)
```

```
Round 8...
4, 3 (7)
Round 9...
4, 4 (8)
Round 10...
3, 4 (7)
Round 11...
5, 5 (10)
Round 12...
5, 2 (7)

Your results:
 2: yes
 3: no
 4: yes
 5: no
 6: yes
 7: yes
 8: yes
 9: yes
10: yes
11: no
12: no
Your total score: 46
```

Hint: use a boolean array to keep track of which values from 2 to 12 the player has already rolled. Each round, check to see if the array element corresponding to that round number is set to false: if so, then that number hasn't been rolled yet, so you can count it.

```java
public class Dice {

    // holds the values of the two dice
    public int firstDie;
    public int secondDie;

    /*
     * Constructs a default pair of dice with a
     * set of random values.
     */
    public Dice() {

        // toss the dice
        tossDice();
    }

    /*
     * Toss the dice by generating 2 random values from
     * 1 to 6.
     */
    public void tossDice() {

        // generate random values for the dice
        firstDie = (int)(Math.random() * 6 + 1);
        secondDie = (int)(Math.random() * 6 + 1);
    }

    /*
     * Return the sum of the two dice.
     */
    public int sum() {
        return firstDie + secondDie;
    }

    /*
     * Returns the value of the dice as a formatted
     * String.
     */
    public String toString() {

        // return the dice as a formatted string
        return String.format("%d, %d", firstDie, secondDie);
    }
}
```

the Dice class

```java
 * Tests the Dice class.
 * Requires: Dice.java
 */
public class TestDice {

    // constants for game data
    public static final int ROUNDS = 12;
    public static final int START = 2;

    public static void main(String[] args) {

        // array to keep track of what values were rolled
        boolean[] rounds = new boolean[ROUNDS-1];

        // construct a pair of dice and display them
        Dice dice = new Dice();

        // display a game start message
        System.out.println("Starting the game!");

        // play 11 rounds (2 to 12)
        for (int i = START; i <= ROUNDS; i++) {

            // display round number
            System.out.printf("Round %d...%n", i);

            // toss the dice and get the sum
            dice.tossDice();
            int sum = dice.sum();

            // display the dice roll
            System.out.printf("%s (%d)%n", dice, sum);

            // if we haven't rolled this number yet,
            // now we have so set it to true
            if (!rounds[sum-START]) {
                rounds[sum-START] = true;
            }
        }

        // game over: calculate score
        int score = 0;
        System.out.println("\nYour results: ");

        // display a list results
        for (int i = 0; i < rounds.length; i++) {

            // if this amt was rolled, add to score
            score += (rounds[i]) ? i + START : 0;

            // display if this value was rolled or not
            String yesNo = (rounds[i]) ? "yes" : "no";
            System.out.printf("%2d: %s%n", i + START, yesNo);
        }

        // display total score
        System.out.printf("Your total score: %d%n", score);
    }
```

}

Practice Problems:

1.

Design a Java program to get a number from user and find the given number is

positive or negative. Display the message.

2.

Write a Java program to calculate a Factorial of a number

3.

Write a Java program to get the age of a person and find the age group of that person as

Age	Category
0-3	Toddlers
4-8	Kids
9-12	Child
13-18	Teens
19 – 40	Adults
41- 60	Matured Adults
61 – 75	Seniors
>76	Super Seniors

4.

Write a program called CozaLozaWoza which prints the numbers 1 to 110, 11 numbers per line. The program shall print "Coza" in place of the numbers which are multiples of 3, "Loza" for multiples of 5, "Woza" for multiples of 7, "CozaLoza" for multiples of 3 and 5, and so on.

The output shall look like:

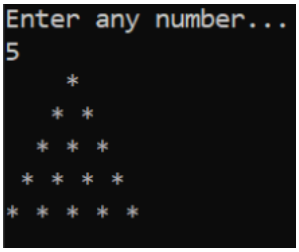
1 2 Coza 4 Loza Coza Woza 8 Coza Loza 11
Coza 13 Woza CozaLoza 16 17 Coza 19 Loza CozaWoza 22
23 oza Loza 26 Coza Woza 29 CozaLoza 31 32 Coza

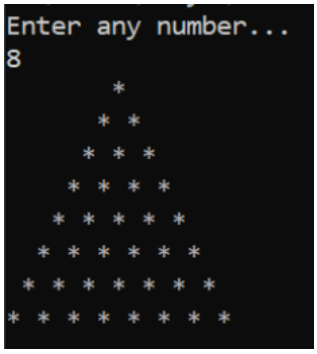
5.

Write a Java code to get a number and find whether it is prime number or not.
(note:Prime number is the number divisible by 1 and itself only)

6.

Design a Java program to print the following pattern for the positive value ‘n’.
Sample Output





7.

Write a Java program that will read in month and day (as numerical value). The program will return the equivalent zodiac sign.

Zodiac Sign	Duration
Aquarius	Jan 20 – Feb 18
Pisces	Feb 19 – Mar 20
Aries	Mar 21 – Apr 19
Taurus	Apr 20 – May 20
Gemini	May 21 – Jun 20
Cancer	Jun 21 – Jul 22
Leo	Jul 23 – Aug 22
Virgo	Aug 23 – Sep 22
Libra	Sep 23 – Oct 22
Scorpio	Oct 23 – Nov 21
Sagittarius	Nov 22 – Dec 21
Capricorn	Dec 22 – Jan 19

Sample output:
Enter month: 6
Enter day: 25
Zodiac sign for June 25 is Cancer

8.
Write Java code statements that accomplish the tasks listed below.

a) Declare an array of integers.
b) Allocate storage to allow 5 integers to be stored in the array.
c) Populate the array with the values: 1, 8, 27, 64, 125
d) Replace the third array element with the value -7.
e) Copy the value of the fifth array element to the first array storage location.
f) Subtract the value stored in the second array storage location from the value stored in the third and store the difference in the fourth array storage location.
g) Compute the sum of the array elements with subscripts 1 to 3.

9.

Design a Java code to receive input for 'n' numbers and sort numbers based on user's choice either ascending or descending. Finally print results

10.

Design a Java program to get a string and do the following in the same program.

a. Get a character. Find the occurrence of the character from right and left side. Display that information separately.

b. Get a positive integer from user and find the character of the index such that should not create run time error.

11.

Design a Java program to get 'n' numbers and a number. Apply the linear search and binary search. Find the best algorithm through the computation and display the result.

12.

Design a Java Program to get a matrix as input and print the transpose of the given matrix.

13.

Develop a Java program to get a matrix and print the lower triangle of the matrix. Apply the necessary conditions if it required.

Sample output:

```
Enter the no.of rows in matrix...
3
Enter no.of cols in the matrix...
3
Enter the elements in the matrix...
1 2 3 4 5 6 7 8 9
Matrix entered by the user...
1 2 3
4 5 6
7 8 9
Lower triangular matrix:
1 0 0
4 5 0
7 8 9
```

14.

Get ‘n’ integer numbers from user and find the count of each unique number. Display the result as number – its count.
Example: 3 - 5, where 3 is the number presented for 5 times)

15.

Write a Java program to get a string from user. Divide the given string into 5 equals parts and make those part as new string. At last, print all information. Apply the necessary conditions

16.

Design a Java Program to represent time as class which contains the following:
a. Hours, minutes and seconds as members.
b. Basic Methods to get input, print the time.
c. Method to find the difference between two times.
Demonstrate these methods through few objects. Also apply the necessary conditions

Sample Output:

```
Starting Time
Enter the hours, minutes and seconds:
08 67 34
Invalid Input
Enter the hours, minutes and seconds:
08 57 34

Ending Time
Enter the hours, minutes and seconds:
10 58 35

Starting Time:
8:57:34
Ending Time:
10:58:35

Time Difference:
2:1:1
```

17.

Design a Java program with your choice of a class to represent any real-world object. Illustrate *constructors* and *this* keyword.

18.

Design a Student class which contains register number, name, marks for three courses, average and result. Define methods to get input, print details, find the average and result (if mark is more than 49 in all courses then Pass, otherwise Fail).
Keep input methods and print methods as public while others are private.
Create array of objects for the Student class and demonstrate those methods

19.

Design a class to store account details of a person like account number, name, account type, available balance and minimum balance. Define methods to get input, display account details, show balance, deposit and withdraw.
Apply the condition while withdraw money from account that the minimum balance to be maintained.
Create a demo class in Java to demonstrate these methods with minimum of 3 objects

20.

Create a class Vehicle which has numOfTyres, model and company as members.
Create constructor to assign values for members, design a method to print members.
Create another class MotorVehicles which inherits Vehicle class.
Members of MotorVehicles class are price, type (Car/Bike).
Define the constructor to assign values of the members and method print which prints values of MotorVehicles members.
Define static members to keep the count of each type of motor vehicles created.
Develop necessary methods for counting and display it at the last.
Apply the necessary mechanisms of inheritance wherever required.
Illustrate these throw a demo class Java program

21.

Write a program that displays all the leap years, seven per line, from 2001 to 2100

22.

Design a class CourseAndMarkswhich can store course name (Course1 or Course2 or Course3) and mark of a student. Include methods to do the following:

I. Getting input for members
II. Display details of members
III. Find the average of each courses which can be stored in a separate variable of theCourseAndMarks class.

Develop a test class to create some objects which are belong to the CourseAndMarks class followed by getting input. Finally display all objects and the average of all three courses. You can include some more fields other than mentioned above which must be relevant for the process.

23.

Design Java programs to accomplish the following:
a. Get details of bank account like account number, name, balance, account type (Savings/Current) and branch name for a few customers and write those into a file.
b. Read those data and display them in a table format. Also, find the average balance of each branch and display it.

24.

Design Java programs to serialize and deserialize the above bank account details. Find the average balance of savings account and current account while deserialize those objects

25.

Design a Java program to store the registration number and name of students in the map. Later, find any name is stored in the register number column too. Print the details if any name is stored in the register number column, otherwise print “All data are correct”.
Also, print the map details.

**********Java Demystified by SSR**********