Lab Assignment 03

Linear Loops

CSE110: Programming Language I

No of Tasks	Points to Score
18	180

[Submit all the Coding Tasks (Task 1 to 15) in the Google Form shared on buX. Submit the Tracing Tasks (Task 16 to 18) handwritten to your Lab Instructors before the next lab.]

1. Write a Java program that prints the following sequence of values using loops:

   18, 27, 36, 45, 54, 63

2. Write a Java program that prints the following sequence of values using loops:

   18, -27, 36, -45, 54, -63

3. Write a Java program which adds all numbers that are multiples of both 7 and 9 up to 600. *[Ans: 2835]*

4. Write a Java program which adds all numbers that are multiples of either 7 or 9 or both up to 600. Ensure that numbers like 63 are added only once in the sum. *[Ans: 42649]*

5. Write a Java program which adds all numbers that are multiples of either 7 or 9 but not both, up to 600. Use only a single loop in your program. *[Ans: 39814]*

6. Write a Java program that will calculate the value of y where the expression of y is as follows (consider n as given):

   $y = 1^2 - 2^2 + 3^2 - 4^2 + 5^2 ..... + n^2$

Given Value of n	Output	Explanation
5	15	y = 1 - 4 + 9 - 16 + 25 = 15
10	-55	y = 1 - 4 + 9 - 16 + 25 - 36 + 49 - 64 + 81 - 100 = -55
20	-210	y = 1 - 4 + 9 - 16 + 25 ……. - 400 = -210

7. Write a Java program that will print all the divisors of a given integer number. Also, print how many divisors are there for that number.

   Given Number:

   6

   Output:

   Divisors of 6:

   1

   2

   3

   6

   Total Divisors: 4

8. Write a Java program that displays the sum of first ***n*** odd natural numbers. Assign the value of n according to your choice.

Given Value:

Number of terms: 5

Expected Output:

The odd numbers are:

1

3

5

7

9

The Sum of odd Natural Numbers up to 5 terms is: 25

9. Write a Java program that will print the first number, the sum of the first 2 numbers, the first 3 numbers, and so on up to the sum of 10 numbers.

Output:

Current Number: 1, Sum: 1
Current Number: 2, Sum: 3
Current Number: 3, Sum: 6
Current Number: 4, Sum: 10
Current Number: 5, Sum: 15
Current Number: 6, Sum: 21
Current Number: 7, Sum: 28
Current Number: 8, Sum: 36
Current Number: 9, Sum: 45
Current Number: 10, Sum: 55

10. Write a Java program that will print all the numbers from 0 to *n* (where n should be assigned according to your choice) which are **divisible by 5 but not divisible by 3**.

Given Value

40

Output

5

10

20

25

35

40

11. Write a program in Java that counts the number of digits in a given integer number.

Hint: You may keep dividing the number by ten and count how many times this can be done until the number becomes 0.

Given Value	Output
7546	Total digits = 4

12. Write a program in Java that prints the individual digits of a given integer number backward (From right to left).

Given Value	Output
32768	8, 6, 7, 2, 3

Hint: First to get the digit from the right side, we can take the remainder of the number using modulus (%) operator i.e. mod 10 to get the rightmost digit and print it. For dropping the last digit, we can perform division by 10 on the number and then continue the same to print the other digits as shown below.
32768 % 10 = 8
32768 / 10 = 3276
Then,
3276 % 10 = 6
3276 / 10 = 327
and so on
327 % 10 = 7
327 / 10 = 32
32 % 10 = 2
32 / 10 = 3
3 % 10 = 3
3 / 10 = 0
Done! When the number becomes 0 you can end your loop.

13. Write a program in Java that prints the individual digits of a given integer number forward (From left to right). ***[You are not allowed to use Math.pow() to solve this problem.]***

Given Value	Output
32768	3, 2, 7, 6, 8

Hint: First, count how many digits. Then calculate 10 to the power that (number of digits) minus 1.Say, 32768 has 5 digits, so you calculate 10 to the power 4 which is 10,000.
Then divide 32,768 by 10,000 and thus you get 3
Take remainder of 32,768 by 10,000 and thus you get 2,768

Then divide 10,000 by 10 to get 1,000

Then divide 2,768 by 1,000 and thus you get 2.
take remainder of 2,768 by 1,000 and thus you get 768
keep going on until there are no more digits left (zero!).

In short:
Part 1: First count digits, say 5 in this case for 32,768
Part 2: Then calculate 10 to the power 4 (5-1), that is 10,000.
Part 3: Then repeat the following three steps:

32,768 / 10,000 = 3
32,768 % 10,000 = 2,768
10,000/10 = 1,000

2,768 / 1,000 = 2
2,768 % 1,000 = 768
1,000/10 = 100

768 / 100 = 7
768 % 100 = 68
100/10 = 10

68 / 10 = 6
68 % 10 = 8
10/10 = 1

8 / 1 = 8
8 % 1 = 0
1/10 = 0

14. Write a Java program that will find out if a given integer number is a prime number or not.

Prime Number: If a number has **only two divisors**, (1 and itself), then it is a prime number. Else, then it is not a prime number.

Given Value:

7

Output:

7 is a prime number

15. Write a Java program that will find out if a given integer number is a perfect number or not.

Perfect Number: A number is said to be a perfect number if the **sum of its divisors**, including 1 but not the number itself is equal to that number.

Given Value:

6

Output:

6 is a perfect number

Explanation:

The divisors of 6 are 1, 2, 3 and 6. Excluding 6, if we take the summation of the remaining divisors we get 1 + 2 + 3 = 6 which is equal to the number itself.

16. Trace the following code, create a tracing table and write the outputs:

```
1  public class Trace1{
2    public static void main(String args[]){
3       int x = 0, p = 0, sum = 0;
4       p = 1;
5       x = 2;
6       double q;
7       sum = 0;
8       while (p < 12){
9          q =  x + p-(sum+7/3)/3.0%2 ;
10         sum = sum + (++x) + (int)q;
11         System.out.println(sum++);
12         if (x > 5){
13            p += 4/2;
14         }
15         else {
16            p += 3%1;
17         }
18      }
19      sum = sum + p;
20      System.out.println(sum);
21   }
22 }
```

17. Trace the following code, create a tracing table and write the outputs:

```
1  public class Trace2 {
2      public static void main(String[] args) {
3          int x = 0, p = 0, sum = 0;
4          p = 1;
5          x = 2;
6          double q = 0.0;
7          sum = 5;
8          while (p < 15) {
9            q = x + p - (sum + (int) (7 / 4)) / 3.0 % 2;
10           sum = sum + x + (int) q;
11           x += 1;
12           System.out.println(sum);
13           if (x > 5) {
14                p += (int) (5 / 2);
15           }
16           else {
17                p += 10 % 3;
18           }
19         }
20         sum = sum + p;
21         System.out.println(sum);
22     }
23 }
```

18. Trace the following code, create a tracing table and write the outputs:

```
1  public class Trace3 {
2    public static void main(String[] args) {
3      int m = 17, n = 13, p = 1, sum = 30;
4      while(0 < p%10){
5         if(m % 10 == 0){
6            sum = sum * m % n + p / n ;
7         }
8         else{
9            if(m % 4 == 0){
10              sum += n + (--m);
11           }
12           else{
13              sum -= m--;
14           }
15        }
16        p+=1;
17        System.out.println(sum);
18     }
19     System.out.println(!(n%13 == 0) && !false || p>10);
20   }
21 }
```