Module 1

1. **The existing system permits the addition of duplicate line items in return orders when a xyz limited company performs a return sale order with reference to a billing document. How can the business prevent adding duplicate line items to a returns selling order?**

```
public class Change
{
  public static int removeDuplicate(int[] arrNumbers, int num)
  {
    if(num == 0 || num == 1)
    {
      return num;
    }
    int[] arrTemporary = new int[num];
    int b = 0;
    for(int a = 0; a < num - 1; a++)
    {
      if(arrNumbers[a] != arrNumbers[a + 1])
      {
        arrTemporary[b++] = arrNumbers[a];
      }
    }
    arrTemporary[b++] = arrNumbers[num - 1];
    for(int a = 0; a < b; a++)
    {
      arrNumbers[a] = arrTemporary[a];
    }
    return b;
  }
```

```
  public static void main(String[] args)
  {
    int[] arrInput = {1, 2, 3, 3, 4, 5, 5, 6, 7, 8};
    int len = arrInput.length;
    len = removeDuplicate(arrInput, len);
    // printing elements
    for(int a = 0; a < len; a++)
    {
      System.out.print(arrInput[a] + " ");
    }
  }
}
```

2. **Gifts for those who were born on February 29th will now be offered by a corporation. Create a Java program that will accept the employee's birth year. Give them jewellery they can wear for years (and leap years) to come if it's a leap year.**

```
import java.util.Scanner;
public class LeapYear {
  public static void main(String[] args){
    int year;
    System.out.println("Enter the Year of birthday ");
    Scanner sc = new Scanner(System.in);
    year = sc.nextInt();

    if (((year % 4 == 0) && (year % 100!= 0)) || (year%400 == 0))
      System.out.println("Specified year is a leap year, gift will be sent");
    else
      System.out.println("Specified year is not a leap year");
  }
}
```

3. **In order to play the lottery, you must choose six numbers from the range of 1 to Write a program to do this for you and generate five sets of entries.**

```
public class Lottery {
  public static void main(String[]args) {
    int setCount = 5;                                  // Number of sets of lucky numbers.
    int setSize = 6;                                   // Number of lucky numbers in the set.
    int range = 49;                                    // Assume selecting integers between 1 and
range.
    int lucky;                                         // Holds a lucky number candidate.
    int luckyCount;                                    // Holds count of lucky numbers in a set.


    for(int i = 0 ; i < setCount ; ++i) {
      int lucky1 = 0;                                  // Lucky numbers for the set of 6.
      int lucky2 = 0;
      int lucky3 = 0;
      int lucky4 = 0;
      int lucky5 = 0;
      int lucky6 = 0;


      luckyCount = 0;                                  // Count of numbers found in the current
set
      while(luckyCount < setSize) {
       // Generate a lucky number between 0 and 48 and add 1:
       lucky = (int)(range*Math.random()) + 1;
        switch(luckyCount) {
          case 0:                                      // It is the first one
            lucky1 = lucky;                            // so just store it
            luckyCount++;                              // and increment the count
            break;
          case 1:                                      // For the second we must
            if(lucky != lucky1) {                      // check that it is different from the first
```

```
          lucky2 = lucky;                    // It is, so store it
          luckyCount++;                      // and increment the count
        }
        break;
      case 2:                                // For the third we check aginst the previous two
        if(lucky != lucky1 && lucky != lucky2) {
          lucky3 = lucky;
          luckyCount++;
        }
        break;
      case 3:                                // Check against the previous three...
        if(lucky != lucky1 && lucky != lucky2 && lucky != lucky3) {
          lucky4 = lucky;
          luckyCount++;
        }
        break;
      case 4:                                // Check against the previous four...
        if(lucky != lucky1 && lucky != lucky2 && lucky != lucky3 && lucky != lucky4) {
          lucky5 = lucky;
          luckyCount++;
        }
        break;
      case 5:                                // Check against the previous five...
        if(lucky != lucky1 && lucky != lucky2 && lucky != lucky3 && lucky != lucky4 && lucky != lucky5) {
          lucky6 = lucky;
          luckyCount++;
        }
        break;
    }
```

```
    }

    System.out.print("\nSet " + (i + 1) + ":");                    // Identify the set

    System.out.print(" " + lucky1 + " " + lucky2  + " " + lucky3  +  // and output the numbers
                     " " + lucky4  + " " + lucky5  + " " + lucky6);
 }
}
}
```

4. To determine how many N individuals can be seated next to one another in a theatre row, develop a Java code.

```
import java.util.*;
public class Main
{
    public static void main(String []args)
    {
      //Take input from the user
      //Create an instance of the Scanner Class
      Scanner sc=new Scanner(System.in);
      //Declare and Initialize the variable
      System.out.println("Enter the number: ");
      int num=sc.nextInt();
      int i=1, fact=1;
      while(i<=num)
      {
         fact=fact*i;
         i++;
      }
      System.out.println("Factorial of the number: "+fact);
    }   }
```

5. **Write a Java Program to Check Palindrome of a number.**

```
import java.util.Scanner;

class ReverseNumber
{
```

```
    public static void main(String[] args)
    {
        int number = 0, reverse = 0, remainder = 0;

        System.out.print("Enter a number:\t");

        Scanner sc = new Scanner(System.in);

        number = sc.nextInt();

        while(number != 0)
        {
            remainder = number % 10;
            reverse = reverse * 10 + remainder;
            number = number/ 10;
        }

        System.out.println("Reversed number is:\t" + reverse);
    }
}
```

Output
Enter a number: 478294
Reversed number is: 492874

6. Given a tank with a predetermined height, radius, and water flow, fill the tank. Ascertain whether the tank will overflow in a specific amount of time. The water flow will be indicated in volume per minute.

```
class Number
{
    // function to calculate the volume of tank
    public static float volume(int radius, int height)
    {
        return ((22 / 7) * radius * radius * height);
    }

    // function to print overflow / filled /
    // underflow accordingly
    public static void check_and_print(double required_time,
                                       double given_time)
    {
        if (required_time < given_time)
            System.out.print( "Overflow" );
        else if (required_time > given_time)
            System.out.print( "Underflow" );
        else
```

```
            System.out.print( "Filled" );
    }

    // driver code
    public static void main(String[] args)
    {
        int radius = 5, // radius of the tank
        height = 10, // height of the tank
        rate_of_flow = 10; // rate of flow of water

        double given_time = 70.0; // time given

        // calculate the required time
        double required_time = volume(radius, height) /
                               rate_of_flow;

        // printing the result
        check_and_print(required_time, given_time);
    }
}
```

7**. Write a program that prompts the user to enter a password.**
Create a boolean variable named valid and set it to true. If any of these tests below fail, set it to true.
Check the password to see if it has at least 8 characters. If it does not, display the message, "Password must have at least 8 characters"
Check the password to see if it consists of only letter and digits. To do this, you will need to loop through all of the characters in the string. A character c is a letter of digit if this expression is true:

('a' <= c && c <= 'z') || ('A' <= c && c <= 'Z') || ('0' <= c && c <= '9')

if this is even not true, break from your loop and display the message, "Password must contain only letter and digits" 5. If valid is still true at the end of the program, display the message, "Password accepted!"

```
import java.util.Scanner;
  public class PasswordVerification {

      public static void main(String[] args) {
          //Creates a scanner
          Scanner sc = new Scanner(System.in);
          boolean valid = false;
          String password;


          //Asks user to enter password
          System.out.print("Please enter password and then hit enter:");
```

```
        password = sc.nextLine();

        //Checks to see if password is at least 8 characters.
        if (password.length()<8)
           {
              valid = false;
              System.out.println("Password must have at least 8 characters");
           }

        //Checks each character to see if it is acceptable.
        for (int i = 0; i < password.length(); i++){
                char c = password.charAt(i);
       if (('a' <= c && c <= 'z') // Checks if it is a lower case letter|| ('A' <= c && c <= 'Z')
//Checks if it is an upper case letter|| ('0' <= c && c <= '9') //Checks to see if it is a digit)
                {

                   valid = true;
                }

                else
                {
     // tells the user that only letters & digits are allowed
    System.out.println("Only letter & digits are acceptable.");
                   valid = false;
                   break;
                }

        }

       // if the password is valid, tell the user it's accepted
        System.out.println("Password Accepted");
        }

  }
```

8. Teachers might have created multiple classes with a few students and thought of just merging them into one big class so it can be easily managed. If all classes will receive the same Announcement and Activities from the Teacher, it makes sense to merge these classes into just one class. Write a java program to combine students of 2 classes into one class.

```
import java.util.Arrays;
import java.util.Scanner;

public class ConcatenateTwoArraysWithoutarraycopy {
```

```
  public static void main(String[] args) {
    int i;
    System.out.println("Enter the required size of the first array: ");
    Scanner reader = new Scanner(System.in);
    int size = reader.nextInt();
    int[] inputFirstArray = new int[size];
    System.out.println("Enter the elements of the array: ");
    for (i = 0; i < size; i++) {
      inputFirstArray[i] = reader.nextInt();
    }
    System.out.println("Enter the required size of the second array: ");
    size = reader.nextInt();
    int[] inputSecondArray = new int[size];
    System.out.println("Enter the elements of the array: ");
    for (i = 0; i < size; i++) {
      inputSecondArray[i] = reader.nextInt();
    }
    int firstLen = inputFirstArray.length;
    int secondLen = inputSecondArray.length;
    int[] mergedArray = new int[firstLen + secondLen];

    int position = 0;
    for (int element : inputFirstArray) {
      mergedArray[position] = element;
      position++;
    }
    for (int element : inputSecondArray) {
      mergedArray[position] = element;
      position++;
    }
    System.out.println("Arrays after merging: ");
    System.out.println(Arrays.toString(mergedArray));
  }
}
```

9. **Calculate the interest earned on $17,000 that was deposited for a year at a rate of 0.07. After a year, calculate the interest and investment value.**

```
public class Interest {
public static void main(String[] args) {
/* Declare the variables. */
double principal; // The value of the investment.
double rate; // The annual interest rate.
double interest; // Interest earned in one year.
/* Do the computations. */
principal = 17000;
rate = 0.07;
interest = principal * rate; // Compute the interest.
principal = principal + interest;
```

```
// Compute value of investment after one year, with interest.
// (Note: The new value replaces the old value of principal.)
/* Output the results. */
System.out.print("The interest earned is $");
System.out.println(interest);
System.out.print("The value of the investment after one year is $");
System.out.println(principal);
} // end of main()
} // end of class Interest
```

10. **Write a Java Program to Find Number of Days in a Month**

```
import java.util.Scanner;

public class DaysinMonth1 {
	private static Scanner sc;
	public static void main(String[] args)
	{
		int month;
		sc = new Scanner(System.in);

		System.out.print(" Please Enter Month Number from 1 to 12 (1 = Jan, and 12
= Dec) : ");
		month = sc.nextInt();

		if (month == 1 || month == 3 || month == 5 || month == 7 || month == 8 ||
month == 10 || month == 12 )
		{
			System.out.println("\n 31 Days in this Month");
		}
		else if ( month == 4 || month == 6 || month == 9 || month == 11 )
		{
			System.out.println("\n 30 Days in this Month");
		}
		else if ( month == 2 )
		{
			System.out.println("\n Either 28 or 29 Days in this Month");
		}
		else
			System.out.println("\n Please enter Valid Number between 1 to 12");
	}
}
```