Assignment 1

1. Write a program to calculate average of all numbers between n1 and n2(eg.100 to 300 Read values of n1 and n2 from user).

```
object Main {
  def main(args: Array[String]): Unit = {

    println("Enter the first number:")
    val n1 = scala.io.StdIn.readInt()
    println("Enter the second number:")
    val n2 =  scala.io.StdIn.readInt()


    var sum = 0
    var count = 0
    for (i <- n1 to n2) {
      sum += i
      count += 1
    }


    val average = sum.toDouble / count


    println(s"The average of the numbers between $n1 and $n2 is:
$average")
  }
}
```

Output

```
Enter the first number:4
Enter the second number:45
The average of the numbers between 4 and 45 is: 24.5
```

2. Write a program to calculate factorial of a number.

```
object factorial{
     def main(args:Array[String])=
     {
          var f=1
          println("Enter number:")
          var n=scala.io.StdIn.readInt()
          for(i<-1 to n.toInt)
          {
               f=f*i
          }
          println(s"Factorial of $n :$f")
     }
}
```

Output

```
Enter number:4
Factorial of 4 :24
```

3. Write a program to read five random numbers and check that random numbers are perfect number or not.

```
def isPerfect(num: Int): Boolean = {
  var sum = 0
  for (i <- 1 until num) {
    if (num % i == 0) {
      sum += i
    }
  }
  sum == num
}

// Read in five random numbers
println("Enter five random numbers:")
val num1 = scala.io.StdIn.readInt()
val num2 = scala.io.StdIn.readInt()
val num3 = scala.io.StdIn.readInt()
val num4 = scala.io.StdIn.readInt()
val num5 = scala.io.StdIn.readInt()
```

```
// Check if each number is a perfect number
if (isPerfect(num1)) println(num1 + " is a perfect number") else
println(num1 + " is not a perfect number")
if (isPerfect(num2)) println(num2 + " is a perfect number") else
println(num2 + " is not a perfect number")
if (isPerfect(num3)) println(num3 + " is a perfect number") else
println(num3 + " is not a perfect number")
if (isPerfect(num4)) println(num4 + " is a perfect number") else
println(num4 + " is not a perfect number")
if (isPerfect(num5)) println(num5 + " is a perfect number") else
println(num5 + " is not a perfect number")
```

Output

```
Enter five random numbers:
4
4
5
6
7

4 is not a perfect number
4 is not a perfect number
5 is not a perfect number
6 is a perfect number
7 is not a perfect number
```

4. Write a program to find second maximum number of four given numbers.

```
object A6
{
def main(args:Array[String])=
{
var a=0;
var b=0;
for(i<- 1 to 4)
{
println("Enter number")
var n=scala.io.StdIn.readInt()
if(n>a)
{
b=a
a=n
}
else if(n>b)
{
```

```
b=n
}

}
println("second maximun number:"+b)
}
}
```

Output

```
Enter the first number:1
Enter the second number:2
Enter the third number:3
Enter the fourth number:4
The second maximum number is: 3
```

5. Write a program to find maximum and minimum of an array.

```
  object A6
{
      def main(args:Array[String])
      {
            var number=Array(1,2,3,4)
            var max=0
            var min=0
            min=number(0)

            for(i<-0 to (number.length)-1)
            {

                  if(max<number(i))
                  {
                        max=number(i)
                  }
                  if(min>number(i))
                  {
                        min=number(i)
                  }
            }
            println("Maximum number in array:- "+max+"Minimum number
in array:- "+min)

      }
}
```

Output

```
Maximum number in array:- 4
Minimum number in array:- 1
```

6. Write a program to calculate determinant of a matrix.

a) for 2x2 matrix

```
object Main {
  def main(args: Array[String]): Unit = {
    // Read the elements of the matrix from the user
    println("Enter the elements of the matrix:")
    println("Enter element (0,0):")
    val a = scala.io.StdIn.readInt()
    println("Enter element (0,1):")
    val b =  scala.io.StdIn.readInt()
    println("Enter element (1,0):")
    val c =  scala.io.StdIn.readInt()
    println("Enter element (1,1):")
    val d =  scala.io.StdIn.readInt()

    // Calculate the determinant of the matrix
    val determinant = a * d - b * c

    // Print the determinant
    println(s"The determinant of the matrix is: $determinant")
  }
}
```

Output

```
Enter the elements of the matrix:
Enter element (0,0):1
Enter element (0,1):2
Enter element (1,0):3
Enter element (1,1):4
The determinant of the matrix is: -2
```

b) for 3x3 matrix

```
object Main {
```

```
  def main(args: Array[String]): Unit = {

    println("Enter the elements of the matrix:")
    println("Enter element (0,0):")
    val a = scala.io.StdIn.readInt()
    println("Enter element (0,1):")
    val b = scala.io.StdIn.readInt()
    println("Enter element (0,2):")
    val c = scala.io.StdIn.readInt()
    println("Enter element (1,0):")
    val d = scala.io.StdIn.readInt()
    println("Enter element (1,1):")
    val e = scala.io.StdIn.readInt()
    println("Enter element (1,2):")
    val f = scala.io.StdIn.readInt()
    println("Enter element (2,0):")
    val g = scala.io.StdIn.readInt()
    println("Enter element (2,1):")
    val h = scala.io.StdIn.readInt()
    println("Enter element (2,2):")
    val i = scala.io.StdIn.readInt()


    val determinant = a * (e * i - f * h) - b * (d * i - f * g) + c
* (d * h - e * g)


    println(s"The determinant of the matrix is: $determinant")
  }
}
```

Output

```
Enter the elements of the matrix:
Enter element (0,0):2
Enter element (0,1):4
Enter element (0,2):5
Enter element (1,0):2
Enter element (1,1):3
Enter element (1,2):4
Enter element (2,0):5
Enter element (2,1):6
Enter element (2,2):7
The determinant of the matrix is: 3
```

Assignment 2

1. Write a program to count uppercase letters in a string and convert it to lowercase and display the new string.

```
object ass2_1{

def countUppercase(s: String): Int = {
  s.count(c => c.isUpper)
}

def main(args:Array[String])={
val s = "HEllo World"
val uppercaseCount = countUppercase(s)
print(uppercaseCount)
print(" ")
print(s.toLowerCase())
}

}
```

Output

```
3 hello world
```

2. Write a program to read a character from user and count the number of occurrences of that character.

```
object ass22 {

def main(args:Array[String])={
val s = "Hello World"
val c = scala.io.StdIn.readChar()



val count = s.filter(_ == c).length
println(s"The character '$c' appears $count times in the string
'$s'")
}

}
```

Output

```
o
The character 'o' appears 2 times in the string 'Hello World'
```

3. Write a program to read two strings. Remove the occurrence of second string in first string.

```
object ass23{

def main(args:Array[String])={
val s1 = scala.io.StdIn.readLine()
val s2 = scala.io.StdIn.readLine()

val s3 = s1.replaceAll(s2, "")
println(s"Original string: '$s1', modified string: '$s3'")

}
}
```

Output

```
hello world
world
Original string: 'hello world', modified string: 'hello '
```

4. Create array of strings and read a string from user. Display all the elements of array containing given string.

```
object ass24{

def main(args:Array[String])={

val arr = Array("apple", "banana", "cherry", "date", "elderberry")
val s = scala.io.StdIn.readLine()
arr.filter(_.contains(s)).foreach(println)
}

}
```

Output

```
rr
cherry
elderberry
```

Assignment 3
List and Set

- Create a list of integers divisible by 3 from List containing numbers from 1 to 50.
- Create two Lists. Merge it and store the sorted in ascending order.
- Write a program to create a list of 1 to 100 numbers. Create second list from first list selecting numbers multiple of 10.
- Create a list of 50 members using function 2n+3. Create second list
  excluding all elements multiple of 7.
- Create a list of even numbers up to 10 and calculate its product.
- Write a program to create list with 10 members using function
  $3n^2+4n+6$
- Write a program to create two sets and find common elements between them.
- Write a program to display largest and smallest element of the Set.
- Write a program to merge two sets and calculate product and average of all elements of the Set.

Assignment 4
Classes and Objects, MAP

- Define a class CurrentAccount (accNo, name, balance, minBalance). Define appropriate constructors and operations withdraw(), deposit(), viewBalance(). Create an object and perform operations.
- Define a class Employee (id, name, salary). Define methods accept() and display(). Display details of employee having maximum salary.
- Create abstract class Order (id, description). Derive two classes
  PurchaseOrder and SalesOrder with members Vendor and Customer. Create object of each PurchaseOrder and SalesOrder. Display the details of each account.
- Write a program to create map with Rollno and FirstName. Print
  all student information with same FirstName.
- Write a user defined functions to convert lowercase letter to uppercase and call the function using Map.

Assignment 3

1. Create a list of integers divisible by 3 from List containing numbers from 1 to 50

```
object ass3_1 {
  def main(args: Array[String]) = {
    val numbers = 1 to 50
    val newlist = numbers.filter(_ % 3 == 0)
    println(newlist)
  }
}
```

Output

```
Vector(3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45, 48)
```

2. Create two Lists. Merge it and store the sorted in ascending order.

```
object ass3_2 {
  def main(args: Array[String]) = {
   val mergedList = (List(1, 11, 5, 7, 9) ++ List(2, 4, 6, 8, 10)).sorted
   println(mergedList)
  }
}
```

Output

```
List(1, 2, 4, 5, 6, 7, 8, 9, 10, 11)
```

3. Write a program to create a list of 1 to 100 numbers. Create second list from first list selecting numbers multiple of 10.

```
object ass3_3 {
  def main(args: Array[String]) = {
  val numbers = 1 to 100
val multiplesOfTen = numbers.filter(_ % 10 == 0)
   println(multiplesOfTen)
```

```
  }
}
```

Output

```
Vector(10, 20, 30, 40, 50, 60, 70, 80, 90, 100)
```

4. Create a list of 50 members using function 2n+3. Create second list excluding all elements multiple of 7.

```
object ass3_4 {
  def main(args: Array[String]) = {
val numbers = (0 until 50).map(2 * _ + 3)
val notMultiplesOfSeven = numbers.filter(_ % 7 != 0)
   println(notMultiplesOfSeven)
  }
}
```

Output

```
Vector(3, 5, 9, 11, 13, 15, 17, 19, 23, 25, 27, 29, 31, 33, 37, 39, 41, 43, 45,
47, 51, 53, 55, 57, 59, 61, 65, 67, 69, 71, 73, 75, 79, 81, 83, 85, 87, 89, 93,
```

95, 97, 99, 101)

5. Create a list of even numbers up to 10 and calculate its product.

```
object ass3_5 {
  def main(args: Array[String]) = {
val numbers = (1 to 10).filter(_ % 2 == 0).product
   println(numbers)
  }
}
```

Output

3840

6. Write a program to create list with 10 members using function $3n^2+4n+6$

```
object ass3_6 {
  def main(args: Array[String]) = {
val numbers = (0 until 10).map(n => 3 * n * n + 4 * n + 6)
```

```
  println(numbers)
 }
}
```

Output

Vector(6, 13, 26, 45, 70, 101, 138, 181, 230, 285)

7. Write a program to create two sets and find common elements between them.

```
object ass3_7 {
 def main(args: Array[String]) = {
val commonElements = Set(1, 3, 5, 7, 2).intersect(Set(2, 4, 6, 8, 10))
 commonElements.foreach(println)
 }
}
```

Output

2
11
8

8. Write a program to display largest and smallest element of the Set.

```
object ass3_8 {
  def main(args: Array[String]) = {
println("Minimum element: " + Set(1, 3, 5, 7, 9).min)
println("Maximum element: " + Set(1, 3, 5, 7, 9,11).max)
  }
}
```

Output

```
Minimum element: 1
Maximum element: 11
```

9. Write a program to merge two sets and calculate product and average of all elements of the Set.

```
object ass3_9 {
  def main(args: Array[String]) = {
   val set1 = Set(1, 3, 5, 7, 9)
    val set2 = Set(2, 4, 6, 8, 10)
```

```
    val mergedSet = set1 ++ set2
    val product = mergedSet.product
    val average = mergedSet.sum.toDouble / mergedSet.size
    println("Merged set: " + mergedSet.mkString(", "))
    println("Product of all elements: " + product)
    println("Average of all elements: " + average)
  }
}
```

Output

```
Merged set: 5, 10, 1, 6, 9, 2, 7, 3, 8, 4
Product of all elements: 3628800
Average of all elements: 5.5
```

Assignment 4

1. Define a class CurrentAccount (accNo, name, balance, minBalance). Define appropriate constructors and operations withdraw(), deposit(), viewBalance(). Create an object and perform operations.

```
class CurrentAccount(val accNo: String, val name: String, var balance:
Double, val minBalance: Double) {
  def withdraw(amount: Double): Unit = {
    if (balance - amount >= minBalance) {
      balance -= amount
      println(s"Withdrawal of $amount successful. New balance: $balance")
    } else {
      println(s"Insufficient balance. Minimum balance should be maintained:
$minBalance")
    }
  }

  def deposit(amount: Double): Unit = {
    balance += amount
    println(s"Deposit of $amount successful. New balance: $balance")
  }

  def viewBalance(): Unit = {
    println(s"Account balance: $balance")
  }
}
```

```
object ass4_1 {
  def main(args: Array[String]): Unit = {
    val account = new CurrentAccount("123456", "John Smith", 1000.0,
500.0)
    account.viewBalance()
    account.withdraw(300.0)
    account.deposit(200.0)
    account.viewBalance()
  }
}
```

Output

```
Account balance: 1000.0
Withdrawal of 300.0 successful. New balance: 700.0
Deposit of 200.0 successful. New balance: 900.0
Account balance: 900.0
```

2. Define a class Employee (id, name, salary). Define methods accept() and display(). Display details of employee having maximum salary.

```
class Employee(var id: Int, var name: String, var salary: Double) {
  def accept(): Unit = {
    println("Enter employee ID:")
```

```
    id = scala.io.StdIn.readLine().toInt
    println("Enter employee name:")
    name = scala.io.StdIn.readLine()
    println("Enter employee salary:")
    salary = scala.io.StdIn.readLine().toDouble
  }

  def display(): Unit = {
    println(s"ID: $id, Name: $name, Salary: $salary")
  }
}

object ass4_2 {
  def main(args: Array[String]): Unit = {
    println("Enter number of employees:")
    val numEmployees = scala.io.StdIn.readLine().toInt
    val employees = List.fill(numEmployees)(new Employee(0, "", 0.0))

    for (employee <- employees) {
      employee.accept()
    }

    val maxSalaryEmployee = employees.maxBy(_.salary)
    maxSalaryEmployee.display()
  }
}
```

Output

```
Enter number of employees:3
Enter employee ID:1
Enter employee name:john
Enter employee salary:400
Enter employee ID:2
Enter employee name:wayne
Enter employee salary:300
Enter employee ID:3
Enter employee name:rick
Enter employee salary:900
ID: 3, Name: rick, Salary: 900.0
```

3. Create abstract class Order (id, description). Derive two classes PurchaseOrder and SalesOrder with members Vendor and Customer. Create object of each PurchaseOrder and SalesOrder. Display the details of each account.

```
abstract class Order(val id: Int, val description: String) {
  def display(): Unit
}

class PurchaseOrder(id: Int, description: String, val vendor: String) extends
Order(id, description) {
  override def display(): Unit = {
    println(s"ID: $id, Description: $description, Vendor: $vendor")
  }
}
```

```
class SalesOrder(id: Int, description: String, val customer: String) extends
Order(id, description) {
  override def display(): Unit = {
    println(s"ID: $id, Description: $description, Customer: $customer")
  }
}

object ass4_3 {
  def main(args: Array[String]): Unit = {
    val purchaseOrder = new PurchaseOrder(1, "ABC Corp", "XYZ Inc.")
    purchaseOrder.display()

    val salesOrder = new SalesOrder(2, "ABC Corp", "John Smith")
    salesOrder.display()
  }
}
```

Output

```
ID: 1, Description: ABC Corp, Vendor: XYZ Inc.
ID: 2, Description: ABC Corp, Customer: John Smith
```

4. Write a program to create map with Rollno and FirstName. Print all student information with same FirstName.

```
object ass4_4 {
  def main(args: Array[String]): Unit = {
    val students = Map(
      1 -> "Alice",
      2 -> "Bob",
      3 -> "Charlie",
      4 -> "Alice",
      5 -> "Alice",
      6 -> "Charlie"
    )

    println("Enter first name:")
    val firstName = scala.io.StdIn.readLine()

    for ((rollNo, name) <- students if name == firstName) {
      println(s"Roll No: $rollNo, First Name: $name")
    }
  }
}
```

Output

```
Enter first name: Alice
Roll No: 5, First Name: Alice
Roll No: 1, First Name: Alice
Roll No: 4, First Name: Alice
```

5. Write a user defined functions to convert lowercase letter to uppercase and call the function using Map.

```
object ass4_5 {
  def toUppercase(c: Char): Char = c.toUpper

  def main(args: Array[String]): Unit = {
    val map = Map('a' -> 'A', 'b' -> 'B', 'c' -> 'C', 'd' -> 'D', 'e' -> 'E', 'f' -> 'F', 'g' ->
'G', 'h' -> 'H', 'i' -> 'I', 'j' -> 'J', 'k' -> 'K', 'l' -> 'L', 'm' -> 'M', 'n' -> 'N', 'o' -> 'O',
'p' -> 'P', 'q' -> 'Q', 'r' -> 'R', 's' -> 'S', 't' -> 'T', 'u' -> 'U', 'v' -> 'V', 'w' -> 'W',
'x' -> 'X', 'y' -> 'Y', 'z' -> 'Z')
    println("Enter a lowercase letter:")
    val c = scala.io.StdIn.readLine().head
    val upperCase = map(c)
    println("the upper case letter is " + upperCase)
  }
}
```

Output

```
Enter a lowercase letter:a

the upper case letter is A
```