Roll No:- _____

Sonopant Dandekar Shikshan Mandali's

Sonopant Dandekar Arts,V.S.Apte Commerce,

M.H.Mehta Science College

DEPARTMENT OF COMPUTER SCIENCE

CERTIFICATE

Certified That Mr./Miss._____

of_____ has satisfactorily completed a course of necessary experiment in _____under my supervision in the SY.BSC Computer Science in the Year 2025 – 2026

Head of Department **Subject Teacher**

Date: / / 2025

INDEX

SR No.	Aim of Practical	Practical Date	Submission Date	Remarks
1.	Write a simple scala program that prints a Welcome message for data scientists.			
2.	Write a scala program to Calculate mean, medians, and mode of a list of numbers, implement basic statistical calculations using sacala collections.			
3.	Write a scala program to Generate a random dataset of 10 numbers and Calculate its variance and standard deviation.			
4.	Write a scala program to Create a dense vector using Breeze and calculate its sum, mean, and dot Product with another vector.			
5.	Write a scala program to generate a random matrix using Breeze and compute its transpose and determinant.			
6.	Write a scala program to slice a Breeze matrix to extract a sub-matrix			
7.	Write a scala program to perform element wise addition, subtraction, multiplication and Division of two breeze matrix.			
8.	Write a Scala program to tokenize and count the frequency of words in a text file			
9.	Write a scala program for handle missing values in a dataset. Replace missing with the column mean.			
10.	Write a scala program to read CSV File and calculate basic statistics for each numeric column. Use the scala-csv library or similar tools.			
11.	Write a scala program to plot a line graph for a dataset showing a trend over time.			
12.	Write a scala program to combine two plots (e.g , scatter and line plot) in a single visualization using breeze-viz			
13.	write a scala program for generate a histogram of a dataset using Breeze-viz experiment with different bin sizes.			
14.	Write a scala program to find the correlation between two lists of numbers. Implement the formula for Pearson correlation coefficient.			
15.	write a scala program to compute frequency distribution and cumulative frequency of dataset.			
16.	write a scala program from sort a dataset by a specific column and extract the top 5 rows.			
17.	Write a scala program to implememnt linear regressionusing breeze .fit a model to a small dataset and predict a value.			

18.	Write a scala program to ompute the euclidean distance between two breeze vector. use it for nearest neighbor classification.			
19.	Write a scala program to calculate the moving average of a time series data using scala collection.			
20.	Write a scala program to Create polynomial features from a dataset. Given a list of numbers(e.g,,[1,2,3]), generate polynomial features up to degree 3. (e.g,..[1,1^2,1^3,2,2^2,2^3])			
21.	Write a scala program to perform basic time series analysis in scala. Generate synthetic time series data (e.g, daily sales over a month).			
22.	Write a scala program to filter rows in sataset where a specific column value exceeds a thresold.			

Date:- _____

Practical No 1

Aim: Write a simple scala program that prints a Welcome message for data scientists.

Note: do practical in notepad++

For checking version of Scala give command in CMD.

Command: scala –version.

```
C:\Users\student>scala -version
Scala code runner version 2.13.16 -- Copyright 2002-2025, LAMP/EPFL and Lightbend, Inc. dba Akka

C:\Users\student>
```

Input:

```
object WelcomMessage1 {
def main(args: Array[String]): Unit={
println("Welcome to the world of Data Science!")
println("Here,we explore data, build models,and derive insights")
}
}
```

Note:

- First create folder in desktop and save file in it.
- Save file with extension .scala with all types.
- Then copy the path of file by going to folder and tap on bar.
- Then give command in CMD.
    1. Cd desktop
    2. Cd folder name
    3. scalac filename.scala
    4. scala filename.scala

```
C:\Users\student>cd Desktop

C:\Users\student\Desktop> cd 65014scala
```

Output:

```
C:\Users\student\Desktop\65014scala>scalac WelcomMessage1.scala

C:\Users\student\Desktop\65014scala>scala WelcomMessage1.scala
Welcome to the world of datascince
Here,we explore data, build models,and derive insights

C:\Users\student\Desktop\65014scala>_
```

Date:_____

Practical No 2

Aim: Write a scala program to Calculate mean, medians, and mode of a list of numbers, implement basic statistical calculations using sacala collections.

Input:

```
object BasicStats {
def main(args: Array[String]): Unit={
val numbers=List(1,2,2,3,4,4,4,5,6)
val mean=calculateMean(numbers)
val median=calculateMedian(numbers)
val mode=calculateMode(numbers)
println(s"Numbers: $numbers")
println(f"Mean: $mean%.2f")
println(s"Median: $median")
println(s"Mode: $mode")
}
def calculateMean(numbers: List[Int]): Double={
numbers.sum.toDouble / numbers.size
}
def calculateMedian(numbers: List[Int]): Double={
val sorted=numbers.sorted
val size=sorted.size
if (size % 2==0){
(sorted(size/2-1)+sorted(size/2).toDouble/2)
}else{
sorted(size/2).toDouble
}
}
def calculateMode(numbers:List[Int]):List[Int] ={
val frequencyMap=numbers.groupBy(identity).view.mapValues(_.size).toMap
val maxFrequency= frequencyMap.values.max
frequencyMap.filter(_._2== maxFrequency).keys.toList
}
}
```

Output:

```
C:\Users\student\Desktop\65014scala>scalac BasicStats.scala

C:\Users\student\Desktop\65014scala>scala BasicStats.scala
Numbers: List(1, 2, 2, 3, 4, 4, 4, 5, 6)
Mean: 3.44
Median: 4.0
Mode: List(4)
```

Date:- _____

Practical No 3

Aim: Write a scala program to Generate a random dataset of 10 numbers and Calculate its variance and standard deviation.

Input:

```
import scala.util.Random
object VarianceStdDevCalculator{
def main(args:Array[String]): Unit={
//Generate random dataset of 10 numbers between 1 to 100
val random=new Random()
val numbers=List.fill(10)(random.nextInt(100)+1)
val mean=calculateMean(numbers)
val variance=calculateVariance(numbers,mean)
val stdDev=math.sqrt(variance)
println(s"Random dataset:$numbers")
println(f",Mean:$mean%.2f")
println(f"Variance:$variance%.2f")
println(f"Standard Deviation:$stdDev%.2f")
}
def calculateMean(numbers:List[Int]): Double={
numbers.sum.toDouble / numbers.size
}
def calculateVariance(numbers:List[Int],mean:Double): Double={
val squaredDiffs=numbers.map(n=>math.pow(n-mean,2))
squaredDiffs.sum/numbers.size
}
}
```

Output:

```
C:\Users\student\Desktop\65014scala>scalac VarianceStdDevCalculator.scala

C:\Users\student\Desktop\65014scala>scala VarianceStdDevCalculator.scala
Random dataset:List(3, 79, 84, 38, 24, 60, 46, 77, 45, 80)
,Mean:53.60
Variance:666.64
Standard Deviation:25.82
```

Date:_____

Practical No 4

Aim: Write a scala program to Create a dense vector using Breeze and calculate its sum, mean, and dot Product with another vector.

Note:

1. Make sure you have SBT Installed.
2. Create a new SBT project

   Mkdir BreezeProject

   cd BreezeProject

   mkdir src\main\scala

```
C:\Users\student\Desktop\65014scala>mkdir BreezeProject

C:\Users\student\Desktop\65014scala>cd BreezeProject

C:\Users\student\Desktop\65014scala\BreezeProject>mkdir src\main\scala

C:\Users\student\Desktop\65014scala\BreezeProject>
```

3. Create a build .sbt file(in notepad++)

   Create File build.sbt file (and then save in BreezeProject>src>main>scala as a build.sbt with all types.

Input:

```
name:="BreezeProject"

version:="0.1"

scalaVersion:="2.13.16"

libraryDependencies ++= Seq(

"org.scalanlp"%%"breeze"%"2.1.0",

"org.scalanlp"%%""breeze-natives"%"2.1.0"//optional for native speed

}
```

4. Write a Scala file using Breeze (in notepad++)

Input:

```
import breeze.linalg._

object BreezeVectorExample{

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

//step1:create a dense vector
```

```
val v1=DenseVector(1.0,2.0,3.0,4.0,5.0)
println(s"Vector V1:$v1")
//step2:Calculate sum
val sum=breeze.linalg.sum(v1)
println(s"Sum of v1:$sum")
//step 3: Calculate mean
val mean=breeze.stats.mean(v1)
println(s"Mean of v1:%$mean")
//step 4:Create another vector and compute dot product
val v2=DenseVector(5.0,4.0,3.0,2.0,1.0)
val dotProduct=v1 dot v2
println(s"Dot product of v1 and v2:$dotProduct")
}
}
```

Output:

```
E:\65014s\BreezeProject1\src\main\scala>sbt run
[info] Updated file E:\65014s\BreezeProject1\src\main\scala\project\build.properties: set sbt.version to 1.10.11
[info] welcome to sbt 1.10.11 (Oracle Corporation Java 1.8.0_171)
[info] loading project definition from E:\65014s\BreezeProject1\src\main\scala\project
[info] loading settings for project scala from build.sbt...
[info] set current project to Breeze Project1 (in build file:/E:/65014s/BreezeProject1/src/main/scala/)
[info] compiling 1 Scala source to E:\65014s\BreezeProject1\src\main\scala\target\scala-2.13\classes ...
[info] running BreezeVectorExample1
Vector V1:DenseVector(1.0, 2.0, 3.0, 4.0, 5.0)
Sum of v1:15.0
Mean of v1:%3.0
Dot product of v1 and v2:35.0
[success] Total time: 8 s, completed 5 Aug, 2025 10:50:54 AM
←[0J
E:\65014s\BreezeProject1\src\main\scala>
```

Date:_____

Practical No 5

Aim: Write a scala program to generate a random matrix using Breeze and compute its transpose and determinant.

```
C:\Users\student>Mkdir BreezeProject

C:\Users\student>cd BreezeProject

C:\Users\student\BreezeProject>mkdir src\main\scala
```

1. Create a build .sbt file(in notepad++)

```
name:="BreezeProject"

version:="0.1"

scalaVersion:="2.13.16"

libraryDependencies ++= Seq(

"org.scalanlp"%%"breeze"%"2.1.0",

"org.scalanlp"%%""breeze-natives"%"2.1.0"//optional for native speed

}
```

2. Write a Scala file using Breeze (in notepad++)

Input:

```
import breeze.linalg._
import breeze.numerics._
object RandomMatrixExample1 {
  def main(args: Array[String]): Unit = {
    val rows = 3
    val cols = 3
    val randomMatrix = DenseMatrix.rand(rows, cols)

    println("Random matrix:")
    println(randomMatrix)

    val transposedMatrix = randomMatrix.t
    println("Transposed matrix:")
    println(transposedMatrix)

    val determinant = det(randomMatrix)
    println(s"Determinant of the random matrix: $determinant")
  }
}
```

Output:

```
E:\65014s\BreezeProject18\src\main\scala>sbt run
[info] Updated file E:\65014s\BreezeProject18\src\main\scala\project\build.properties: set sbt.version to 1.10.11
[info] welcome to sbt 1.10.11 (Oracle Corporation Java 1.8.0_171)
[info] loading project definition from E:\65014s\BreezeProject18\src\main\scala\project
[info] loading settings for project scala from build.sbt...
[info] set current project to Breeze Project18 (in build file:/E:/65014s/BreezeProject18/src/main/scala/)
[info] compiling 1 Scala source to E:\65014s\BreezeProject18\src\main\scala\target\scala-2.13\classes ...
[info] running RandomMatrixExample1
Random matrix:
0.5812377392133563   0.7175577399068689    0.781118917164858
0.6683751594541827   0.9373189484453164    0.9734871049321954
0.09043030647811534  0.014162641938025633  0.08456167173112372
Transposed matrix:
0.5812377392133563  0.6683751594541827  0.09043030647811534
0.7175577399068689  0.9373189484453164  0.014162641938025633
0.781118917164858   0.9734871049321954  0.08456167173112372
Aug 05, 2025 10:43:38 AM dev.ludovic.netlib.lapack.InstanceBuilder initializeNative
WARNING: Failed to load implementation from:dev.ludovic.netlib.lapack.JNILAPACK
Determinant of the random matrix: 0.001853811544457246
[success] Total time: 14 s, completed 5 Aug, 2025 10:43:39 AM
←[0J
E:\65014s\BreezeProject18\src\main\scala>
```

Date: _____

Practical No 6

Aim: Write a scala program to slice a Breeze matrix to extract a sub-matrix

---

```
C:\Users\student>mkdir BreezeProject18
C:\Users\student>cd Breezeproject18
C:\Users\student\BreezeProject18>mkdir src\main\scala
C:\Users\student\BreezeProject18>_
```

1. Create a build .sbt file(in notepad++)

   Input:

```
name:="BreezeProject"
version:="0.1"
scalaVersion:="2.13.16"
libraryDependencies ++= Seq(
"org.scalanlp"%%"breeze"%"2.1.0",
"org.scalanlp"%%"breeze-natives"%"2.1.0"//optional for native speed
}
```

2. Write a Scala file using Breeze (in notepad++)

   Input:

```
import breeze.linalg._
object MatrixOperations {
def main(args: Array[String]): Unit={
val matrix = DenseMatrix((1.0,2.0,3.0,4.0),
                (5.0,6.0,7.0,8.0),
                (9.0,10.0,11.0,12.0),
                (13.0,14.0,15.0,16.0)
)
println("Original Matrix:")
println(matrix)

val subMatrix=matrix(1 to 2,0 to 1)
println("\nExtract Sub-Matrix:")
```

```
println(subMatrix)
val rowSums=sum(subMatrix(*,::))
println("\nRow sums of sub-matrix:")
println(rowSums)
val colSums = sum(subMatrix(::,*))
println("\nColumn Sum of sub-matrix:")
println(colSums)
}
}
```

Output:

```
E:\65014s\BreezeProject18\src\main\scala>sbt run
[info] welcome to sbt 1.10.11 (Oracle Corporation Java 1.8.0_171)
[info] loading project definition from E:\65014s\BreezeProject18\src\main\scala\project
[info] loading settings for project scala from build.sbt...
[info] set current project to BreezeProject18 (in build file:/E:/65014s/BreezeProject18/src/main/scala/)
[info] compiling 1 Scala source to E:\65014s\BreezeProject18\src\main\scala\target\scala-2.13\classes ...
[info] running MatrixOperations
Original Matrix:
1.0   2.0   3.0   4.0
5.0   6.0   7.0   8.0
9.0   10.0  11.0  12.0
13.0  14.0  15.0  16.0

Extract Sub-Matrix:
5.0  6.0
9.0  10.0

Row sums of sub-matrix:
DenseVector(11.0, 19.0)

Column Sumd of sub-matrix:
Transpose(DenseVector(14.0, 16.0))
[success] Total time: 7 s, completed 5 Aug, 2025 11:44:43 AM
←[0J
E:\65014s\BreezeProject18\src\main\scala>
```

Date:_____

Practical No 7

Aim: Write a scala program to perform element wise addition, subtraction, multiplication and Division of two breeze matrix.

```
C:\65014scala>mkdir BreezeProject24

C:\65014scala>cd BreezeProject24

C:\65014scala\BreezeProject24>mkdir src\main\scala

C:\65014scala\BreezeProject24>
```

SBT File:

```
name:="BreezeProject24"
version:="0.1"
scalaVersion:="2.13.16"
libraryDependencies ++= Seq(
"org.scalanlp"%%"breeze"%"2.1.0",
"org.scalanlp"%%"breeze-natives"%"2.1.0"//optional for native speed
}
```

Input:

```
import breeze.linalg._
import breeze.numerics._//Required for :* and :/
import breeze.math.MutableInnerProductModule._
//Important import for implicit ops
object MatrixElementWiseOps {
def main(args: Array[String]): Unit = {
val matA=DenseMatrix((1.0,2.0), (3.0,4.0))
val matB=DenseMatrix((5.0,6.0), (7.0,8.0))
val addition=matA + matB
val subtraction=matA - matB
val multiplication=matA.mapPairs { case ((i,j),v) => v * matB(i,j) }
val division=matA.mapPairs { case ((i,j),v) => v / matB(i,j) }
println("Matrix A:")
println(matA)
println("\nMatrix B:")
```

```
println(matB)
println("\nElement-wise Addition (A+B):")
println(addition)
println("\nElement-wise Subtraction (A-B):")
println(subtraction)
println("\nElement-wise Multiplication (A :* B):")
println(multiplication)
println("\nElement-wise Division (A :/ B):")
println(division)
}
}
```

Output:

```
C:\65014scala\BreezeProject24\src\main\scala>sbt run
[info] Updated file C:\65014scala\BreezeProject24\src\main\scala\project\build.properties: set sbt.version to 1.10.11
[info] welcome to sbt 1.10.11 (Oracle Corporation Java 1.8.0_461)
[info] loading project definition from C:\65014scala\BreezeProject24\src\main\scala\project
[info] loading settings for project scala from build.sbt...
[info] set current project to BreezeProject24 (in build file:/C:/65014scala/BreezeProject24/src/main/scala/)
[info] compiling 1 Scala source to C:\65014scala\BreezeProject24\src\main\scala\target\scala-2.13\classes ...
[info] running MatrixElementWiseOps
Matrix A:
1.0  2.0
3.0  4.0

Matrix B:
5.0  6.0
7.0  8.0

Element-wise Addition (A+B):
6.0   8.0
10.0  12.0

Element-wise Subtraction (A-B):
-4.0  -4.0
-4.0  -4.0

Element-wise Multiplication (A :* B):
5.0   12.0
21.0  32.0

Element-wise Division (A :/ B):
0.2                 0.3333333333333333
0.42857142857142855  0.5
[success] Total time: 5 s, completed Aug 12, 2025 10:50:17 AM
←[0J
C:\65014scala\BreezeProject24\src\main\scala>
```

Date:_____

Practical No 8

Aim: Write a Scala program to tokenize and count the frequency of words in a text file

---

Scala File:

```
import scala.io.Source
object WordCountShort {
def main(args: Array[String]): Unit = {
val text=Source.fromFile("sample.txt").mkString
val words=text.toLowerCase.split("\\W+").filter(_.nonEmpty)
val freq=words.groupBy(identity).mapValues(_.length).toSeq.sortBy(-_._2)
freq.foreach{case(word,count)=>println(s"$word:$count")}
}
}
```

Txt file:

scala is powerful.Scala is scalable and fun.Scala runs on the JVM.

Hello world! This is a test.

Hello again,world.This is Simple Test.

Output:

```
C:\65014scala\BreezeProject24\src\main\scala>scala WordCountShort.scala
warning: 1 deprecation (since 2.13.0); re-run with -deprecation for details
is:4
scala:3
this:2
world:2
hello:2
test:2
runs:1
a:1
scalable:1
on:1
powerful:1
simple:1
fun:1
and:1
the:1
jvm:1
again:1
```

Date:_____

Practical No 9

Aim: Write a scala program for handle missing values in a dataset. Replace missing with the column mean.

```
C:\65014scala>mkdir BreezeProject18

C:\65014scala>cd BreezeProject18

C:\65014scala\BreezeProject18>mkdir src\main\scala

C:\65014scala\BreezeProject18>
```

SBT File:

```
name:="BreezeProject18"
version:="0.1"
scalaVersion:="2.13.16"
libraryDependencies +="org.scalanlp"%%"breeze"%"2.1.0"
```

Input:

```
import breeze.linalg._
import breeze.stats.mean
import scala.Double.NaN
object HandleMissingValues{
def main(args:Array[String]):Unit={
val data=DenseMatrix(
(1.0,Double.NaN,3.0),
(4.0,5.0,Double.NaN),
(Double.NaN,8.0,9.0)
)
println("Original Matrix with Missing valuses(NaN):")
println(data)
//replace NaN with column mean
for(j<-0 until data.cols){
val col=data(::,j)
val valid=col.toArray.filter(!_.isNaN)
val colMean=if(valid.nonEmpty)mean(DenseVector(valid))else 0.0
```

```
for(i<-0 until data.rows){
if(data(i,j).isNaN){
data(i,j)=colMean
}
}
}
println("\nMatrix after Replacing NaN with Column Mean:")
println(data)
}
}
```

Output:

```
C:\65014scala\BreezeProject18\src\main\scala>sbt run
[info] welcome to sbt 1.10.11 (Oracle Corporation Java 1.8.0_461)
[info] loading project definition from C:\65014scala\BreezeProject18\src\main\scala\project
[info] loading settings for project scala from build.sbt...
[info] set current project to BreezeProject18 (in build file:/C:/65014scala/BreezeProject18/src/main/scala/)
[info] compiling 1 Scala source to C:\65014scala\BreezeProject18\src\main\scala\target\scala-2.13\classes ...
[info] running HandleMissingValues
Original Matrix with Missing valuses(NaN):
1.0  NaN  3.0
4.0  5.0  NaN
NaN  8.0  9.0

Matrix after Replacing NaN with Column Mean:
1.0  6.5  3.0
4.0  5.0  6.0
2.5  8.0  9.0
[success] Total time: 4 s, completed Aug 25, 2025 10:41:19 AM
←[0J
C:\65014scala\BreezeProject18\src\main\scala>
```

Date:_____

Practical No 10

Aim: Write a scala program to read CSV File and calculate basic statistics for each numeric column. Use the scala-csv library or similar tools.

```
C:\65014scala>mkdir BreezeProject

C:\65014scala>cd BreezeProject

C:\65014scala\BreezeProject>src\main\scala
The system cannot find the path specified.

C:\65014scala\BreezeProject>mkdir src\main\scala

C:\65014scala\BreezeProject>_
```

CSV File: data.csv(Save in Breezeproject Folder)

Name,Age,Salary

Alice,25,50000

Bob,30,60000

Charlie,28,55000

David,35,65000

Eve,40,70000

SBT file:

```
name :="BreezeProject"
version:="0.1"
scalaVersion:="2.13.16"
libraryDependencies +="com.github.tototoshi"%%"scala-csv"%"1.3.10"
```

Scala File:

Input:

```
import com.github.tototoshi.csv._
import java.io.File
object CsvStatistics{
def main(args:Array[String]): Unit={
//Read the CSV File
val reader=CSVReader.open(new File("data.csv"))
```

```
val data=reader.allWithHeaders()
//initialize statistics
val ageStats=data.map(row=>row("Age").toInt)
val salaryStats=data.map(row=>row("Salary").toDouble)
//calculate statistics for age
val ageMean=ageStats.sum.toDouble/ageStats.size
val ageMin=ageStats.min
val ageMax=ageStats.max
//calculate statistics for Salary
val salaryMean=salaryStats.sum/salaryStats.size
val salaryMin=salaryStats.min
val salaryMax=salaryStats.max
//print statistics
println(s"Age-Mean:$ageMean,Min:$ageMin, Max:$ageMax")
println(s"Salary-Mean:$salaryMean,Min:$salaryMin, Max:$salaryMax")
//close the reader
reader.close()
}
}
```

Output:

```
C:\65014scala\BreezeProject\src\main\scala>sbt run
[info] welcome to sbt 1.10.11 (Oracle Corporation Java 1.8.0_461)
[info] loading project definition from C:\65014scala\BreezeProject\src\main\scala\project
[info] loading settings for project scala from build.sbt...
[info] set current project to BreezeProject (in build file:/C:/65014scala/BreezeProject/src/main/scala/)
[info] compiling 1 Scala source to C:\65014scala\BreezeProject\src\main\scala\target\scala-2.13\classes ...
[info] running CsvStatistics
Age-Mean:31.6,Min:25, Max:40
Salary-Mean:60000.0,Min:50000.0, Max:70000.0
[success] Total time: 3 s, completed Aug 25, 2025 11:43:37 AM
←[0J
C:\65014scala\BreezeProject\src\main\scala>
```

Date:_____

Practical No 11

Aim: Write a scala program to plot a line graph for a dataset showing a trend over time.

```
C:\65014scala>mkdir breezeproject24

C:\65014scala>cd breezeproject24

C:\65014scala\breezeproject24>_
```

SBT File:

library Dependencies +="org-scalanlp"%%"breeze"%"1.1"

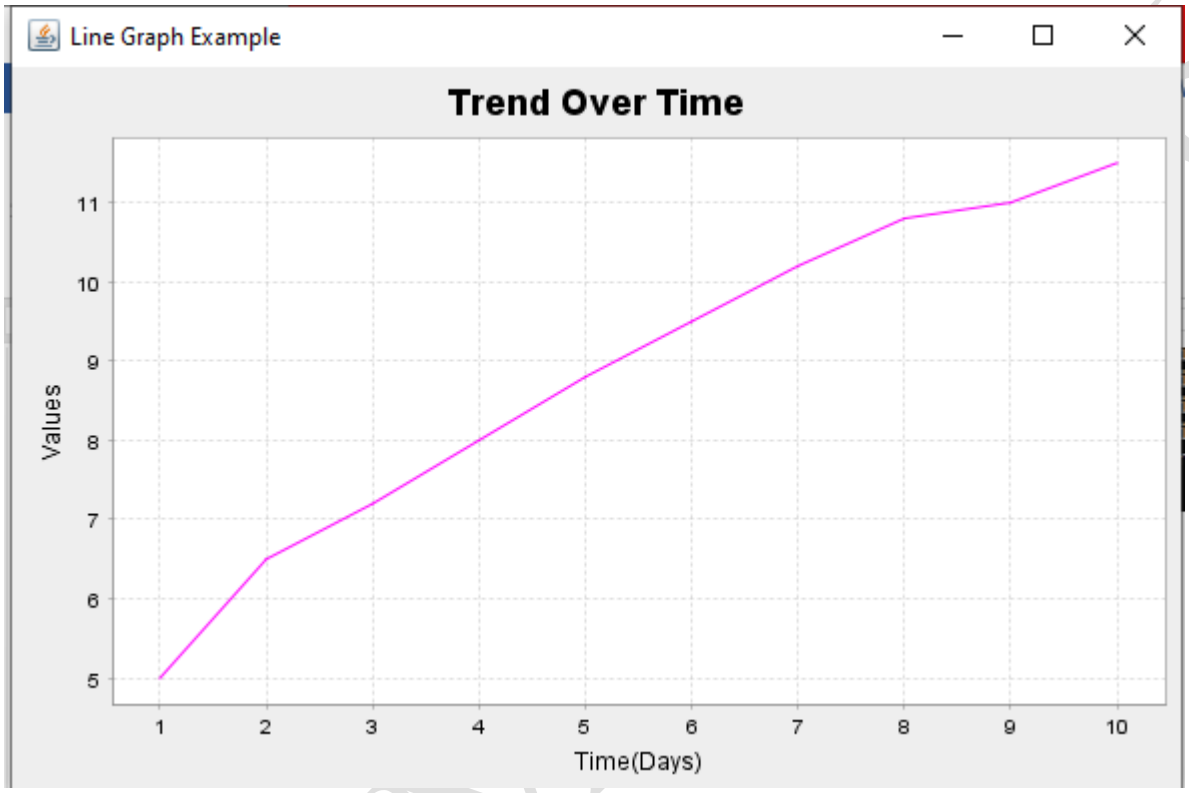
library Dependencies +="org-scalanlp"%%"breeze-viz"%"1.1"

Input:

```
import breeze.linalg._
import breeze.plot._
object LineGraphShort{
def main(args:Array[String]): Unit={
//Example dataset:time(x-axis)and values(y-axis)
val time=linspace(1.0,10.0,10)
val values=DenseVector(5.0,6.5,7.2,8.0,8.8,9.5,10.2,10.8,11.0,11.5)
//create figure and plot
val fig=Figure("Line Graph Example")
val plt=fig.subplot(0)
plt +=plot(time,values,colorcode="m")//magenta line plt.xlabel="Time(Days)"
plt.ylabel="Values"
plt.title="Trend Over Time"
//Keep the window open
println("Press ENTER to close the graph window...")
scala.io.StdIn.readLine()
}
}
```

Output:

```
C:\65014scala\breezeproject24>sbt run
[info] welcome to sbt 1.10.11 (Oracle Corporation Java 1.8.0_461)
[info] loading project definition from C:\65014scala\breezeproject24\project
[info] loading settings for project breezeproject24 from build.sbt...
[info] set current project to breezeproject24 (in build file:/C:/65014scala/breezeproject24/)
[info] compiling 1 Scala source to C:\65014scala\breezeproject24\target\scala-2.12\classes ...
[info] running LineGraphShort
Press ENTER to close the graph window...
```



Date:_____

Practical No 12

Aim: Write a scala program to combine two plots (e.g , scatter and line plot) in a single visualization using breeze-viz

---

```
C:\65014scala\breezeproject24>mkdir BreezePlotProject

C:\65014scala\breezeproject24>cd BreezePlotProject

C:\65014scala\breezeproject24\BreezePlotProject>_
```

SBT file:

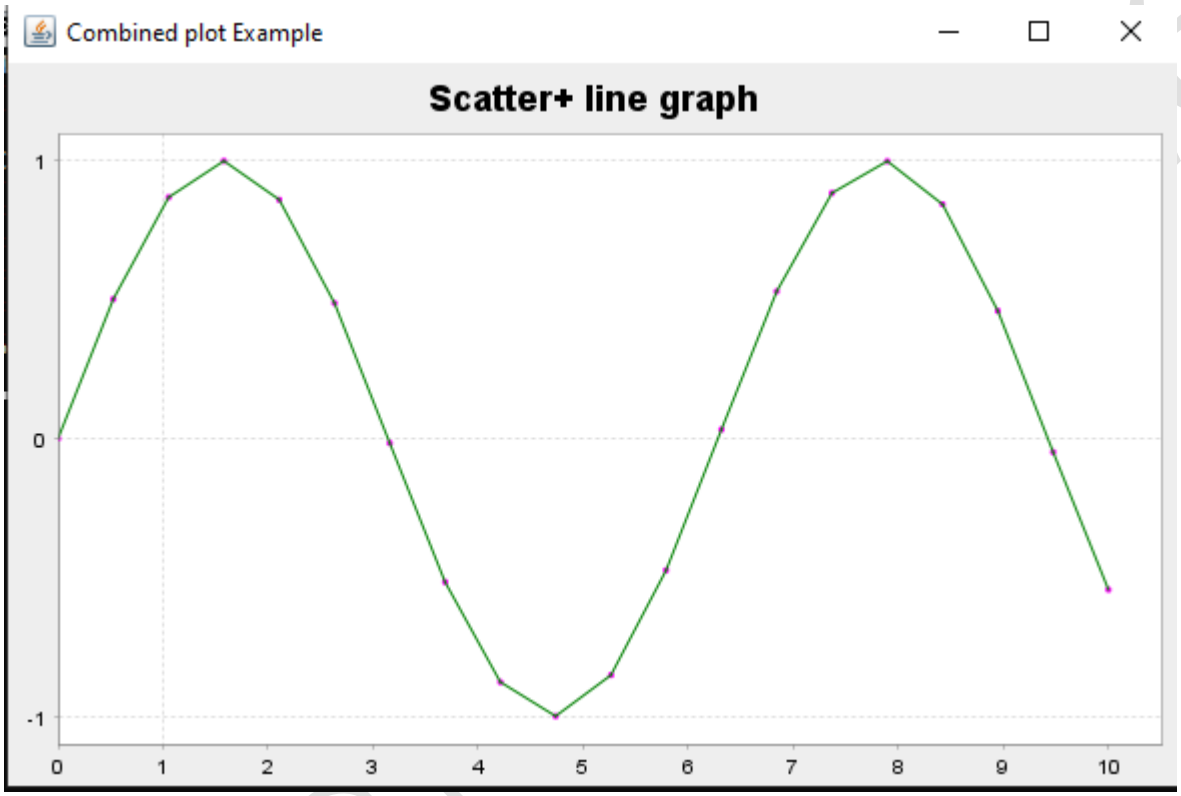
```
name :="BreezePlotProject"
version :="0.1"
scalaVersion :="2.13.16"
libraryDependencies ++=Seq(
"org.scalanlp"%%"breeze"%"2.1.0"
"org.scalanlp"%%"breeze-viz"%"2.1.0"
)
```

Input:

```
import breeze.linalg._
import breeze.plot._
object CombinedPlotExample{
def main(args:Array[String]): Unit={
//Example dataset:
val x=linspace(0.0,10.0,20)
val y=x.map(v =>math.sin(v))
//create figure
val fig=Figure("Combined plot Example")
val plt=fig.subplot(0)
//Line plot (blue sine curve)
plt +=plot(x,y,colorcode="g")
//scatter plot
```

```
plt +=plot(x,y,'.',colorcode="m")
plt.title="Scatter+ line graph"
println("Press ENTER to close the graph window...")
scala.io.StdIn.readLine()
}
}
```

Output:



Date:_____

Practical No 13

Aim: write a scala program for generate a histogram of a dataset using Breeze-viz experiment with different bin sizes.

```
Microsoft Windows [Version 10.0.19045.6216]
(c) Microsoft Corporation. All rights reserved.

C:\65014scala>mkdir BreezeProject2

C:\65014scala>cd BreezeProject2

C:\65014scala\BreezeProject2>_
```

SBT file:

```
libraryDependencies ++=Seq(
"org.scalanlp"%%"breeze"%"2.1.0",
"org.scalanlp"%%"breeze-viz"%"2.1.0"
)
```

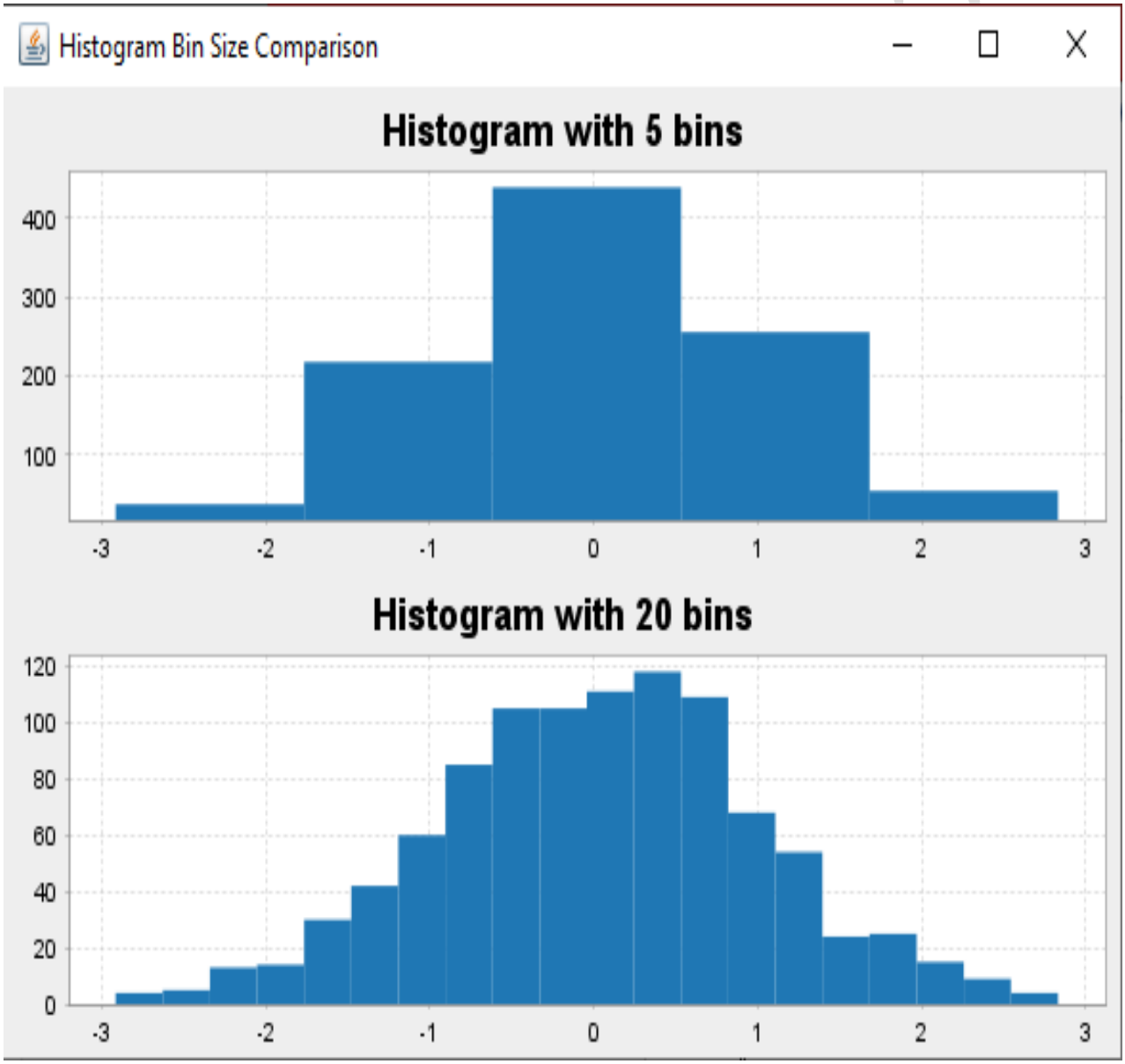
Scala File:

Input:

```
import breeze.linalg._
import breeze.plot._
import breeze.stats.distributions.Rand
object HistogramBins{
def main(args: Array[String]):Unit={
//Generate randaon dataset :1000 samples from normal distributions
val data=DenseVector.rand(1000,Rand.gaussian)
//create figure with two subplots
val fig=Figure ("Histogram Bin Size Comparison")
val plt1=fig.subplot(2,1,0)//top subplot
val plt2=fig.subplot(2,1,1)//bottom subplot
//Histogram with 5 bins
plt1 +=hist(data, bins=5)
plt1.title="Histogram with 5 bins"
```

```
//Histogram with 20 bins
plt2 +=hist(data, bins=20)
plt2.title="Histogram with 20 bins"
//keep window open till Enter is pressed.
println("press Enter to close...")
scala.io.StdIn.readLine()
}
}
```

Output:



Date:_____

Practical No 14

Aim: Write a scala program to find the correlation between two lists of numbers. Implement the formula for Pearson correlation coefficient.

According to Karl Pearson, "Coefficient of Correlation is calculated by dividing the sum of products of deviations from their respective means by their number of pairs and their standard deviations."

Formula: $r = [n(\sum xy) - (\sum x)(\sum y)] / \sqrt{[n(\sum x^2) - (\sum x)^2][n(\sum y^2) - (\sum y)^2]}$

- **r**: is the Pearson correlation coefficient.
- **n**: is the number of data points (pairs) in the dataset.
- **∑xy**: is the sum of the product of the x and y values for each pair.
- **∑x**: is the sum of all x values.
- **∑y**: is the sum of all y values.
- **∑x²**: is the sum of the squares of all x values.
- **∑y²**: is the sum of the squares of all y values.

Input:

```
object PearsonCorrelation{
def main(args: Array[String]):Unit={
val listX=List(1.0,2.0,3.0,4.0,5.0)
val listY=List(2.0,3.0,4.0,5.0,6.0)
val correlation=calculatePearsonCorrelation(listX,listY)
println(s"Pearson correlation coefficient:$correlation")
}
def calculatePearsonCorrelation(x:List[Double],y:List[Double]):Double={
val n=x.length
val sumX=x.sum
val sumY=y.sum
val sumXY=(x zip y).map {case(xi,yi)=>xi*yi}.sum
val sumX2=x.map(xi=>xi*xi).sum
val sumY2=y.map(yi=>yi*yi).sum
val numerator=n*sumXY-sumX*sumY
val denominator=Math.sqrt((n*sumX2-sumX*sumX)*(n*sumY2-sumY*sumY))
numerator/denominator
}
}
```

Output:

```
C:\65014scala>scalac PearsonCorrelation.scala

C:\65014scala>scala PearsonCorrelation.scala
Pearson correlation coefficient:1.0
```

Date:_____

Practical No 15

Aim: write a scala program to compute frequency distribution and cumulative frequency of dataset.

A frequency distribution is a way of organizing data to show how often each value or group of values occurs. It helps in understanding the pattern or trend in a dataset by showing the number of times each data point or range appears.
Cumulative frequency is the running total of frequencies up to a certain point in a frequency distribution. It shows how many data points fall **below or within** a particular value, helping to understand the distribution and spread of data.

Input:

```
object FrequencyCumulative{
def main(args: Array[String]):Unit={
val data=List(2,4,2,6,4,4,8,6,2,10,8,6,4)
val freq=data.groupBy(identity).mapValues(_.size).toSeq.sortBy(_._1)
val cum=freq.map(_._2).scanLeft(0)(_+_).tail
println("Val Freq Cum");freq.zip(cum).foreach{case((v,f),c)=>println(s"$v $f $c")}
}
}
```

Output:

```
C:\65014scala>scalac FrequencyCumulative.scala
warning: 1 deprecation (since 2.13.0); re-run with -deprecation for details
1 warning

C:\65014scala>scala FrequencyCumulative.scala
warning: 1 deprecation (since 2.13.0); re-run with -deprecation for details
Val Freq Cum
2 3 3
4 4 7
6 3 10
8 2 12
10 1 13
```

Date:_____

Practical No 16

Aim: write a scala program from sort a dataset by a specific column and extract the top 5 rows.

Input:

```
object sortAndTop5{
def main(args: Array[String]): Unit={
//sample dataset:id,name,score
val data=List(
(1,"Alice",85),
(2,"Bob",92),
(3,"Charlie",78),
(4,"David",90),
(5,"Eve",88),
(6,"Frank",95),
(7,"Grace",80)
)
//sort by score descending and take top 5
val top5=data.sortBy(-_._3).take(5)
println("Top 5 rows sorted by score:")
top5.foreach{case(id,name,score)=>
println(s"$id,$name,$score")
}
}
}
```

Note: sortBy(-_._3)sorts the list by the third element (score) in descending order.

Output:

```
C:\65014scala>scalac sortAndTop5.scala

C:\65014scala>scala sortAndTop5.scala
Top 5 rows sorted by score:
6,Frank,95
2,Bob,92
4,David,90
5,Eve,88
1,Alice,85
```

Date:_____

Practical No 17

Aim: Write a scala program to implememnt linear regression using breeze .fit a model to a small dataset and predict a value.

```
C:\65014scala>mkdir breezeproject3

C:\65014scala>cd breezeproject3

C:\65014scala\breezeproject3>
```

Linear regression is a statistical method used to model the relationship between two variables by fitting a straight line (called the regression line) to the observed data. It shows how the dependent variable changes as the independent variable changes. The basic form of the linear regression equation is:
y=a+bx

Sbt file:
libraryDependencies+="org.scalanlp"%%"breeze"%"2.1.0"
Scala file:
Input:

```
import breeze.linalg._
import breeze.stats.regression.leastSquares
object LinearRegressionBreeze{
def main(args: Array[String]): Unit={
//Design matrix X:first column is intercept (1.0),second is a feature x
val X=DenseMatrix((1.0,1.0),(1.0,2.0),(1.0,3.0),(1.0,4.0),(1.0,5.0))
val y=DenseVector(2.0,4.1,6.0,8.1,10.2)//Target values
//fit linear regression model
val model=leastSquares(X,y)
val intercept=model.coefficients(0)
val slope=model.coefficients(1)
println(f"Flitted Model:y=$intercept%.2f+$slope%.2f*x")
//predict for x=6.0
val xNew=6.0
val yPred=intercept+slope*xNew
println(f"prediction for x=$xNew%.1f:y=$yPred%.2f")
}
}
```

Output:

```
C:\65014scala\breezeproject3>sbt run
[info] welcome to sbt 1.10.11 (Oracle Corporation Java 1.8.0_461)
[info] loading project definition from C:\65014scala\breezeproject3\project
[info] loading settings for project breezeproject3 from build.sbt...
[info] set current project to breezeproject3 (in build file:/C:/65014scala/breezeproject3/)
[info] compiling 1 Scala source to C:\65014scala\breezeproject3\target\scala-2.12\classes ...
[info] running LinearRegressionBreeze
Sep 16, 2025 11:08:06 AM dev.ludovic.netlib.lapack.InstanceBuilder initializeNative
WARNING: Failed to load implementation from:dev.ludovic.netlib.lapack.JNILAPACK
Flitted Model:y=-0.04+2.04*x
prediction for x=6.0:y=12.20
[success] Total time: 5 s, completed Sep 16, 2025 11:08:06 AM
←[0J
C:\65014scala\breezeproject3>
```

Date:_____

Practical No 18

Aim: Write a scala program to compute the euclidean distance between two breeze vector. use it for nearest neighbor classification.

Euclidean Distance is defined as the distance between two points in Euclidean space. To find the distance between two points, the length of the line segment that connects the two points should be measured.
Euclidean distance is like **measuring the straightest and shortest path between two points**.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Where,
- d is Euclidean Distance,
- (x1, y1, z1) is the Coordinate of the first point,
- (x2, y2, z2) is the Coordinate of the second point.

```
E:\65014s>mkdir breezeproject24

E:\65014s>cd breezeproject24
```

Sbt file:

```
name := "EuclideanDistanceExample"

version := "0.1"

scalaVersion := "2.13.16"

libraryDependencies += "org.scalanlp" %% "breeze" % "2.1.0"
```

Input:

```
import breeze.linalg._

import breeze.numerics._

object EuclideanDistanceExample {

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

val vector1 = DenseVector(3.0,4.0,5.0)

val vector2 = DenseVector(1.0,1.0,1.0)

val distance = euclideanDistance(vector1,vector2)

println(s"The Euclidean Distance between the vectors is:$distance")

}
```

```
def euclideanDistance(v1:DenseVector[Double],v2:DenseVector[Double]):

Double = {

norm(v1 - v2)

}

}
```

Output:

```
E:\65014s\breezeproject24>sbt run
[info] Updated file E:\65014s\breezeproject24\project\build.properties: set sbt.version to 1.10.11
[info] welcome to sbt 1.10.11 (Oracle Corporation Java 1.8.0_171)
[info] loading project definition from E:\65014s\breezeproject24\project
[info] loading settings for project breezeproject24 from build.sbt...
[info] set current project to EuclideanDistanceExample (in build file:/E:/65014s/breezeproject24/)
[info] compiling 1 Scala source to E:\65014s\breezeproject24\target\scala-2.13\classes ...
[info] running EuclideanDistanceExample
The Euclidean Distance between the vectors is:5.385164807134504
[success] Total time: 9 s, completed 20 Sep, 2025 10:34:44 AM
←[0J
```

Date:_____

Practical No 19

Aim: Write a scala program to calculate the moving average of a time series data using scala collection.

Input:

```
object MovingAverageCalculator{
def movingAverage(data: Seq[Double],windowSize:Int):Seq[Double]={
require(windowSize>0,"Window size must be positive.")
if (data.length<windowSize) Seq.empty
else data.sliding(windowSize).map(window => window.sum/windowSize).toSeq
}
def main(args: Array[String]): Unit={
val timeSeriesData=Seq(10.0,20.0,30.0,40.0,50.0,60.0)
val windowSize=3
val result=movingAverage(timeSeriesData, windowSize)
println(s"Time series data:$timeSeriesData")
println(s" Moving average with window size $windowSize:$result")
}
}
```

Output:

```
E:\65014s>scalac MovingAverageCalculator.scala

E:\65014s>scala MovingAverageCalculator.scala
Time series data:List(10.0, 20.0, 30.0, 40.0, 50.0, 60.0)
 Moving average with window size 3:List(20.0, 30.0, 40.0, 50.0)
```

Date:_____

Practical No 20

Aim: Write a scala program to Create polynomial features from a dataset. Given a list of numbers(e.g,,[1,2,3]), generate polynomial features up to degree 3. (e.g,..[1,1^2,1^3,2,2^2,2^3])

Input:

```
object polynomialFeatures{
def main(args: Array[String]): Unit={
//input list of numbers
val inputData=List(1,2,3)
val degree=3
//Generate polynomial feature
val polynomialFeatures=inputData.flatMap{
num=>(1 to degree).map(pow=>math.pow(num,pow).toInt)
}
//print result
println(s"Input Data:$inputData")
println(s"Polynomial Features up to degree $degree:$polynomialFeatures")
}
}
```

Output:

```
E:\65014s>scalac polynomialFeatures.scala

E:\65014s>scala polynomialFeatures.scala
Input Data:List(1, 2, 3)
Polynomial Features up to degree 3:List(1, 1, 1, 2, 4, 8, 3, 9, 27)
```

Date:_____

Practical No 21

Aim: Write scala program to perform basic time series analysis in scala. Generate synthetic time series data (e.g, daily sales over a month).

Input:

```
import scala.util.Random
object TimeSeriesAnalysis {
def main(args: Array[String]): Unit={
val days=30
val rand=new Random()
//Generate synthetic daily series data(between 50 and 150)
val salesData=(1 to days).map(day=>(day,50+rand.nextInt(100)))
println("===Daily sales Data===")
salesData.foreach { case(day,sales)=>println(s"Day $day->$sales") }
//Exact only sales values
val sales=salesData.map(_._2)
//summary statistics
val avg=sales.sum.toDouble / sales.length
val max=sales.max
val min=sales.min
println("\n===Summary Statistics===")
println(f"Average Sales:$avg%.2f")
println(s"Max Sales:$max")
println(s"Min Sales:$min")
//Daily difference
val diffs=sales.sliding(2).collect{case Seq(prev,curr)=>curr-prev}.toSeq
println("\n===Daily Diffrences(Change from previous day)===")
diffs.zipWithIndex.foreach{case(d,i)=>
println(s"Day ${i+2}: Change=$d")
}
//Moving Average(window size=3)
val movingAvg=sales.sliding(3).map(window=>window.sum.toDouble / window.size).toSeq
println("\n===3-Day moving Average===")
movingAvg.zipWithIndex.foreach{case(ma,i)=>
println(f"Days ${i+1}-${i+3}:Moving Avg=$ma%.2f")
}
}
}
```

Output:

```
C:\65014scala>scalac TimeSeriesAnalysis.scala

C:\65014scala>scala TimeSeriesAnalysis.scala
===Daily sales Data===
Day 1->54
Day 2->65
Day 3->62
Day 4->70
Day 5->104
Day 6->94
Day 7->103
Day 8->140
Day 9->56
Day 10->85
Day 11->111
Day 12->147
Day 13->138
Day 14->53
Day 15->81
Day 16->146
Day 17->109
Day 18->101
Day 19->137
Day 20->73
Day 21->88
Day 22->104
Day 23->118
Day 24->108
Day 25->76
Day 26->56
Day 27->74
Day 28->60
Day 29->73
Day 30->121

===Summary Statistics===
Average Sales:93.57
Max Sales:147
Min Sales:53
```

```
===Daily Diffrences(Change from previous day)===
Day 2: Change=11
Day 3: Change=-3
Day 4: Change=8
Day 5: Change=34
Day 6: Change=-10
Day 7: Change=9
Day 8: Change=37
Day 9: Change=-84
Day 10: Change=29
Day 11: Change=26
Day 12: Change=36
Day 13: Change=-9
Day 14: Change=-85
Day 15: Change=28
Day 16: Change=65
Day 17: Change=-37
Day 18: Change=-8
Day 19: Change=36
Day 20: Change=-64
Day 21: Change=15
Day 22: Change=16
Day 23: Change=14
Day 24: Change=-10
Day 25: Change=-32
Day 26: Change=-20
Day 27: Change=18
Day 28: Change=-14
Day 29: Change=13
Day 30: Change=48
```

```
===3-Day moving Average===
Days 1-3:Moving Avg=60.33
Days 2-4:Moving Avg=65.67
Days 3-5:Moving Avg=78.67
Days 4-6:Moving Avg=89.33
Days 5-7:Moving Avg=100.33
Days 6-8:Moving Avg=112.33
Days 7-9:Moving Avg=99.67
Days 8-10:Moving Avg=93.67
Days 9-11:Moving Avg=84.00
Days 10-12:Moving Avg=114.33
Days 11-13:Moving Avg=132.00
Days 12-14:Moving Avg=112.67
Days 13-15:Moving Avg=90.67
Days 14-16:Moving Avg=93.33
Days 15-17:Moving Avg=112.00
Days 16-18:Moving Avg=118.67
Days 17-19:Moving Avg=115.67
Days 18-20:Moving Avg=103.67
Days 19-21:Moving Avg=99.33
Days 20-22:Moving Avg=88.33
Days 21-23:Moving Avg=103.33
Days 22-24:Moving Avg=110.00
Days 23-25:Moving Avg=100.67
Days 24-26:Moving Avg=80.00
Days 25-27:Moving Avg=68.67
Days 26-28:Moving Avg=63.33
Days 27-29:Moving Avg=69.00
Days 28-30:Moving Avg=84.67
```

Date:_____

Practical No 22

Aim: Write a scala program to filter rows in sataset where a specific column value exceeds a thresold.

Input:

```
object FilterRowsExample{
case class Person(name:String,age:Int,salary:Double)
def main(args:Array[String]): Unit={
//sample dataset
val data=Seq(
Person("Alice",25,50000),
Person("Bob",30,60000),
Person("Charlie",22,45000),
Person("David",35,75000),
Person("Eve",28,52000),
)
//threshold for filterating(e.g,salary>55000)
val threshold=55000.0
//filter rows
val filtered=data.filter(person=>person.salary>threshold)
println(s"Rows where salary>$threshold:")
filtered.foreach(p=>println(s"Name:${p.name},Age:${p.age},salary:${p.salary}"))
}
}
```

Output:

```
C:\65014scala>scalac FilterRowsExample.scala

C:\65014scala>scala FilterRowsExample.scala
Rows where salary>55000.0:
Name:Bob,Age:30,salary:60000.0
Name:David,Age:35,salary:75000.0

C:\65014scala>
```