**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Control Structure\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Que:1)Write a programme to calculate the average of all numbers between n1 and n2(eg 100 to 300) read value n1 and n2 from user**

import scala.io.StdIn.\_;

object Avg

{

def main(args:Array[String])

{

printf("Enter two numbers and find the average between them=\n");

var n1=readInt();

var n2=readInt();

var count=0;

var sum=0;

for(i<-n1 to n2)

{

sum+=i;

count+=1;

}

printf("avg between two no is="+(sum/count));

}

}

**Que :2)write a programme to calculate the factorial of a number.**

import scala.io.StdIn.\_;

object Fact

{

def main(args:Array[String])=

{

println("Enter two numbers for calculare the factorial");

var n=readInt();

var fact=1;

for(i<-1 to n)

{

fact=fact\*i;

}

println("factorial of given no is=="+fact)

}

}

**Que:3) Write a programme to read five random numbers and check that random number is perfect or not**.

object Ass

{

def main(args:Array[String])

{

for (i<-1 to 5)

{

val r=scala.util.Random;

checkperfect(r.nextInt(100))

}

}

def checkperfect(n:Int)

{

var sum=0;

for(i<-1 to n/2)

{

if(n%i==0)

sum+=i;

}

if (sum==n)

println(n+"is a perfect number");

else

println(n+" is not a perfect number");

}

}

**Que:4)write a programme to find second maximum number of four given no.**

import scala.io.StdIn.\_;

import scala.util.Sorting.\_;

object SecondMax{

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

var a,b,c,d,max,max2=0;

println("Enter 1st number");

a=readInt();

println("Enter 2nd number");

b=readInt();

println("Enter 3rd number");

c=readInt();

println("Enter 4th number");

d=readInt();

if(a>b && a>c && a>d)

a=Int.MinValue;

else if(b>a && b>c && b>d)

b=Int.MinValue;

else if(c>b && c>a && c>d)

c=Int.MinValue;

else if(d>b && d>c && d>a)

d=Int.MinValue;

if(a>b && a>c && a>d)

println(s"Second MAX is $a");

else if(b>a && b>c && b>d)

println(s"Second MAX is $b");

else if(c>b && c>a && c>d)

println(s"Second MAX is $c");

else if(d>b && d>c && d>a)

println(s"Second MAX is $d");

}

}

**5) Write a program to calculate sum of prime numbers between 1 to 100**

import scala.math.\_;

object SumPrime{

def checkPrime(p:Int):Boolean=

{

var sum=0;var i=2;

var flag:Boolean=true;

while(i<=sqrt(p).toInt && flag)

{

if(p%i==0)

flag=false;

i=i+1;

}

flag;

}

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

var sum=2;

for(i<-3 to 100)

{

if(checkPrime(i))

sum=sum+i;

}

println(s"Sum of prime from 1 to 100 $sum");

}

}

**OR**

object SumPrime

{

def main(args:Array[String])=

{

var primesum=0;

for(i<-2 to 100)

{

var sum=0;

for(j<-2 to i)

if(i%j==0)

sum=sum+j;

if(sum==i)

{

primesum=primesum+i;

println("no is prime no",+i);

println("addition is",primesum);

}

}

}

}

**6)Write a program to read an integer from user and convert it to binary and octal using user defined functions**.

import scala.io.StdIn.\_;

object BinOct{

def Binary(n:Int):String={

var buffer:String="";

var no=n;

while(no>0)

{

buffer=(no%2).toString+buffer;

no=no/2;

}

buffer="0"+buffer;

buffer;

}

def Octal(n:Int):String={

var buffer:String="";

var no=n;

while(no>0)

{

buffer=(no%8).toString+buffer;

no=no/8;

}

buffer=buffer;

buffer;

}

def main(sr:Array[String])=

{

println("Enter number")

var n=readInt();

var bin:String=Binary(n);

var oct:String=Octal(n);

println(s"Binary :$bin || Octal :$oct");

}

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ARRAYS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Que 1)Write a programme to find maximum and minimum of an array.**

import scala.io.StdIn.\_;

object ArrayMaxMin

{

def main(args:Array[String])=

{

println("Enter The No Of Araay Elements");

var n=readInt();

var RollNo=new Array[Int](n);

println("Enter Array Element");

for(i<-0 to n-1)

{

RollNo(i)=readInt();

}

println("Array Element Are");

for(i<-0 to n-1)

{

println(RollNo(i)+"");

}

Max(RollNo);

Min(RollNo);

}

def Max(RollNo:Array[Int])=

{

var Max=0;

Max=RollNo(0);

for(i<-1 to RollNo.length-1)

{

if(RollNo(i)>Max)

Max=RollNo(i);

}

println("Maximum Array Element is:"+Max);

}

def Min(RollNo:Array[Int])=

{

var Min=0;

Min=RollNo(0);

for(i<-1 to RollNo.length-1)

{

if(RollNo(i)<Min)

Min=RollNo(i);

}

println("Minimum Array Element is:"+Min);

}

}

**Que 2)Write a programme to calculate the transpose of a matrix**

import scala.io.StdIn.\_;

object ArrayTranspose

{

println("Enter the no of rows and column");

var row=readInt();

var col=readInt();

def main(args:Array[String])=

{

var Matrix=Array.ofDim[Int](row,col)

println("Enter Matrix Elements");

for(i<-0 to row-1)

{

for(j<-0 to col-1)

{

Matrix(i)(j)=readInt();

}

}

println("Matrix Element Are");

for(i<-0 to row-1)

{

for(j<-0 to col-1)

{

print(Matrix(i)(j)+" ");

}

println();

}

var MatrixT=Array.ofDim[Int](col,row);

for(i<-0 to row-1)

{

for(j<-0 to col-1)

{

MatrixT(j)(i)=Matrix(i)(j);

}

}

println("Transpose Matrix is=");

for(i<-0 to col-1)

{

for(j<-0 to row-1)

{

print(MatrixT(i)(j)+" ");

}

println();

}

}

}

**3)Write a programme to calculate Determinant of Matrix**

import scala.io.StdIn.\_;

object Determinant1

{

def main(args:Array[String]){

println("Enter size of square matrix")

var n=readInt()

var Matrix=Array.ofDim[Int](n,n)

println("Enter matrix elements")

for(i<-0 to n-1)

{

for(j<-0 to n-1)

{

Matrix(i)(j)=readInt()

}

}

println(s"Matrix elements are")

for(i<-0 to n-1)

{

for(j<-0 to n-1)

{

print(Matrix(i)(j)+" ");

}

println();

}

var d=determinant(Matrix,n);

println("Determinant of matrix is",d);

}

def determinant(Matrix:Array[Array[Int]],n:Int):Int={

var det=0;

var s=1;

var Temp=Array.ofDim[Int](n,n)

if(n==1)

return(Matrix(0)(0));

else

{

for(c<-0 to n-1)

{

var row=0

var col=0

for( i<-0 to n-1)

{

for(j<-0 to n-1)

{

Temp(i)(j)=0;

if(i!=0 && j!=c)

{

Temp(row)(col)=Matrix(i)(j);

if(col<(n-2))

col=col+1

else

{

col=0;

row=row+1

}

}

}

}

det=det+s\*(Matrix(0)(c)\*determinant(Temp,n-1));

s=s\*(-1)

}

}

return det;

}

}

**4)Write a programme to check if matrix is upper triangular or not.**

import scala.io.StdIn.\_;

import scala.util.control.Breaks.\_

object UpperTraingularMatrix

{

println("Enter size of square matrix")

var size=readInt()

def main(args:Array[String])=

{

var Matrix=Array.ofDim[Int](size,size)

println("Enter matrix elements")

for(i<-0 to size-1)

{

for(j<-0 to size-1)

{

Matrix(i)(j)=readInt()

}

}

println("Matrix elements are")

for(i<-0 to size-1)

{

for(j<-0 to size-1)

{

print(Matrix(i)(j)+" ");

}

println();

}

for(i<-0 to size-1)

{

for(j<-0 to size-1)

{

if((i>j)&&(Matrix(i)(j)!=0))

{

println("Non-Upper Traingular Matrix")

break;

}

}

}

println("Upper Traingular Matrix")

}

}

**5)Write a programme to sort a matrix using insertion sort.**

import scala.io.StdIn.\_;

import scala.util.Random;

import scala.math.\_;

object SortMat{

def InsertionSort(A:Array[Int],n:Int)={

var i,j=0;

while(i<n)

{

var key=A(i);

j=i-1;

while(j>=0 && key<A(j))

{

A(j+1)=A(j);

j=j-1;

}

A(j+1)=key;

i=i+1;

}

}

def SortIt(A:Array[Array[Int]],r:Int,c:Int)={

var temp=new Array[Int](r\*c);

var arrtrav=0;

for(i<-0 until r)

{

for(j<-0 until c)

{

temp(arrtrav)=A(i)(j);

arrtrav=arrtrav+1;

}

}

InsertionSort(temp,arrtrav);

arrtrav=0;

for(i<-0 until r)

{

for(j<-0 until c)

{

A(i)(j)=temp(arrtrav);

arrtrav=arrtrav+1;

}

}

}

def DisplayMat(A:Array[Array[Int]],r:Int,c:Int)=

{

println("---------------ARRAY-----------------");

for(i<-0 until r)

{

for(j<-0 until c)

{

print(s"${A(i)(j)} ");

}

println("");

}

}

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

var row,column=0;

println("Enter number of row and column");

row=readInt();

column=readInt();

var A=Array.ofDim[Int](row,column);

println("Enter elements");

for(i<-0 until row)

for(j<-0 until column)

A(i)(j)=abs(Random.nextInt());

DisplayMat(A,row,column);

SortIt(A,row,column);

DisplayMat(A,row,column);

}

}

**6)Write a programme for multiplication of two matrices**

import scala.io.StdIn.\_

import Array.\_

object ArrayMultiplication

{

println("Enter no of Rows and Columns of first matrix");

var row=readInt()

var column=readInt()

println("Enter no of Rows and Columns of second matrix");

var row2=readInt()

var column2=readInt()

def main(args:Array[String])

{

var matrix1 = ofDim[Int](row,column)

println("Enter first matrix elements")

for(i<-0 to row-1)

{

for(j<-0 to column-1)

{

matrix1(i)(j)=readInt()

}

}

println("Matrix elements are")

for(i<-0 to row-1)

{

for(j<-0 to column-1)

{

print(matrix1(i)(j)+" ");

}

println();

}

var matrix2 = ofDim[Int](row2,column2)

println("Enter second matrix elements")

for(i<-0 to row2-1)

{

for(j<-0 to column2-1)

{

matrix2(i)(j)=readInt()

}

}

println("Matrix elements are")

for(i<-0 to row2-1)

{

for(j<-0 to column2-1)

{

print(matrix2(i)(j)+" ");

}

println();

}

Multiplication(matrix1,matrix2)

}

def Multiplication(m1:Array[Array[Int]],m2:Array[Array[Int]])

{

if(column!=row2)

{

println("Dimension of matrix should be compatible(columns of First matrix must equal to rows of second matrix)")

}

else

{

var sum=0

var Product=ofDim[Int](row,column2)

for(i<-0 to row-1)

{

for(j<-0 to column2-1)

{

for(k<-0 to column-1)

{

sum=sum+m1(i)(k)\*m2(k)(j)

}

Product(i)(j)=sum;

sum=0;

}

}

println("Product Matrix=")

for(i<-0 to row-1)

{

for(j<-0 to column2-1)

{

print(Product(i)(j)+" ");

}

println();

}

}

}

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*STRING\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**1)Write a programme to count uppercase letters in a string and convert it into lower case & print new String.**

import scala.io.StdIn.\_;

object UppToLow

{

def main(args:Array[String])=

{

println("Enter String");

var str=readLine();

var i:Int=0;

var count:Int=0

var len:Int= str.length();

for(i<-0 to len-1)

{

if(str.charAt(i)>='A'&& str.charAt(i)<='Z')

{

count=count+1;

}

}

println("No of uppercase letter"+count)

var newstr=""

for(i<-0 to len-1)

{

if(str.charAt(i)>='A'&& str.charAt(i)<='Z')

{

var c=(str.charAt(i)+32).toChar;

newstr=newstr+(c);

}

if(str.charAt(i)>='a'&& str.charAt(i)<='z')

{

newstr=newstr+(str.charAt(i));

}

}

println("New String:"+newstr)

}

}

**2)Write a programme to read a character from user and count the no of occurences of character**

import scala.io.StdIn.\_;

object OccurrenceOfChar

{

def main(args:Array[String])=

{

println("Enter String");

var str:String=readLine();

var i:Int=0;

var count=0

println("Enter character to count occurance")

var c=readChar()

var len=str.length();

for(i<-0 to len-1)

{

if(str.charAt(i)==c)

{

count=count+1

}

}

println("Occurrence of "+c+ " in "+str+" is=" +count)

}

}

**3)Write a programme to read two string.Remove the occurrence of second string in first string**.

import scala.io.StdIn.\_;

object DeleteSubstring

{

def main(args:Array[String])=

{

println("Enter first String");

var str1:String=readLine();

println("Enter second String");

var str2:String=readLine();

var str3=""

if(str1.contains(str2)==true)

{

str3=str1.replace(str2,"")

}

println(str3);

}

}

**4)Create Array of String and read a string from user.Display all the elements of array containing given String**

import scala.io.StdIn.\_

object ArrayOfString

{

def main(args:Array[String])=

{

println("Enter string")

var str=readLine()

println(str)

var strarr=str.split(" ")

println("Array Elements are")

for(i<-strarr)

println(i)

}

}

**\*\*\*\*\*\*\*\*\*\*\*CLASS & OBJECTS\*\*\*\*\*\*\*\*\***

**1)Define a class Current Account(AccNo,name,balance,minBalance).Define appropriate Constructers & operation withdraw(),deposite(),viewBalance().Create an object and perform operations**.

import scala.util.control.Breaks.\_

class CurrentAccount()

{

var accno:Int=0;

var name:String=null;

var bal:Double=0.0;

var minbal:Double=500.0;

def Accept()

{

println("enter acno,name");

accno=scala.io.StdIn.readInt();

name=scala.io.StdIn.readLine();

}

def Withdraw()

{

println("enter the amount to withdraw:");

var wamt:Double=scala.io.StdIn.readDouble();

if(bal-wamt>minbal)

bal=bal-wamt;

else

println("insuffient balance please deposit money");

}

def Deposit()

{

println("enter the amount to deposit:");

var damt:Double=scala.io.StdIn.readDouble();

bal=bal+damt;

}

def ViewBalance()

{

println("Currernt AC balnce is:"+bal);

}

}

object Programe1

{

def main(args: Array[String])

{

println("enter how many customers:");

var n:Int=scala.io.StdIn.readInt();

var arr=new Array[CurrentAccount](n);

for(i<- 0 to n-1)

{

arr(i)=new CurrentAccount();

arr(i).Accept();

}

println("enter customer no for operations:");

var no:Int=scala.io.StdIn.readInt();

var ch:Int=0;

while(ch!=4){

println("\n");

println("1 for Deposit\t 2 for Withdraw\t 3 for viewbalance\t 4 for exit");

println("enter your choice:");

var ch:Int=scala.io.StdIn.readInt();

ch match

{

case 1 => arr(no).Deposit();

case 2 => arr(no).Withdraw();

case 3 => arr(no).ViewBalance();

case 4 => break;

}

}

}

}

**2)Define class Employee(id,name,salary).Define Method accept() and Display().Display detail of Employee Having Maximum salary.**

import scala.io.StdIn.\_;

class Employee{

private var id:Int=0;

private var name:String="";

private var salary:Double=0.0;

def accept()={

println("Enter id");

id=readInt();

println("Enter name");

name=readLine();

println("Enter salary");

salary=readDouble();

}

def display()={

//println(s"ID:$id Name:$name Salary:$salary");

println(id+"\t"+name+"\t"+salary)

}

def getSalary:Double=salary;

}

object Programe2

{

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

println("Enter no.s of employee");

var n=readInt();

var p=new Array[Employee](n);

for(i<-0 to n-1)

{

p(i)=new Employee;

p(i).accept();

}

println("ID \t Name \t Salary")

for(i<-0 to n-1)

p(i).display();

var maxSal:Double=p(0).getSalary;

var maxIndex=0;

for(i<-0 to n-1)

{

if(p(i).getSalary>maxSal)

{

maxSal=p(i).getSalary;

maxIndex=i;

}

}

println("MAX SALARY EMPLOYEE");

println("ID \t Name \t Salary")

p(maxIndex).display();

}

}

**3) Create abstract class order(id,description).Derive two classes purchase order & amp sales order with members Vendors and customer.create object of each purches ordr and sales order.Display details of each account.**

abstract class Order{

protected var id:Int;

protected var description:String;

}

class Vendor(private var vid:Int,private var vname:String,private var address:String){

def VId:Int=vid

def VName:String=vname

def VAddress:String=address

}

class Customer(private var cid:Int,private var cname:String,private var address:String){

def CId:Int=cid

def CName:String=cname

def CAddress:String=address

}

class PurchaseOrder(var id:Int,var description:String,var vid:Int,var vname:String,var vadd:String,var cid:Int,var cname:String,var cadd:String) extends Order{

var vendor=new Vendor(vid,vname,vadd)

var customer=new Customer(cid,cname,cadd)

def showDetails()={

println("---------Purchase Order---------")

println(s"OrderId:$id\nDescription:$description")

println("\*"\*50)

println("Customer Details:")

println(s"Name:${customer.CName}\nCId:${customer.CId}\nAddress:${customer.CAddress}")

println("\*"\*50)

println("Vendor Details:")

println(s"Name:${vendor.VName}\nVId:${vendor.VId}\nAddress:${vendor.VAddress}")

println("\*"\*50)

}

}

class SalesOrder(var id:Int,var description:String,var vid:Int,var vname:String,var vadd:String,var cid:Int,var cname:String,var cadd:String) extends Order{

var vendor=new Vendor(vid,vname,vadd)

var customer=new Customer(cid,cname,cadd)

def showDetails()={

println("---------Sales Order---------")

println(s"OrderId:$id\nDescription:$description")

println("~"\*40)

println("Vendor Details:")

println(s"Name:${vendor.VName}\nVId:${vendor.VId}\nAddress:${vendor.VAddress}")

println("~"\*40)

println("Customer Details:")

println(s"Name:${customer.CName}\nCId:${customer.CId}\nAddress:${customer.CAddress}")

println("~"\*40)

}

}

object OrderDet{

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

var pod1=new PurchaseOrder(120232,"Garments",987453,"John.M","New High,Los Angeles\nUSA",256487,"Aryan.L","Andheri East,Navi Mumbai,\nMaharashtra,India");

pod1.showDetails()

var sod1=new SalesOrder(168474,"Decoratings",998745,"Kiran.K","Homy Stores,Hyderabad\nTelangana India",269745,"Vivek.R","Pimpri,Pune,\nMaharashtra,India");

sod1.showDetails()

}

}

**4) Create abstract class Shape with abstract functions volume() and display(). Extend two classes Cube and Cylinder from it. Create object of Cube and Cylinder, Calculate volume of each and display it.**

abstract class shape(s:Double,r:Double,h:Double)

{

def volume(); //abstract method

def display(); //abstract method

}

class cube(s:Double,r:Double,h:Double) extends shape(s:Double,r:Double,h:Double)

{

var vol:Double=0;

def volume()

{

vol=s\*s\*s;

}

def display()

{

volume()

println("Volume of cube is "+vol)

}

}

class cylender(s:Double,r:Double,h:Double) extends shape(s:Double,r:Double,h:Double)

{

var vol:Double=0;

def volume()

{

vol = 3.14\*(r\*r)\*h;

}

def display()

{

volume()

println("Volume of cylinder is "+vol)

}

}

object Abstract

{

def main(args: Array[String])

{

var s:Double=0;

var r:Double=0;

var h:Double=0;

println(" Enter the length of side for Cube:");

s=scala.io.StdIn.readDouble();

println(" Enter the radius and height of Cylinder:");

r=scala.io.StdIn.readDouble();

h=scala.io.StdIn.readDouble();

var ob1=new cube(s,r,h);

ob1.volume();

ob1.display();

var ob2=new cylender(s,r,h);

ob2.volume();

ob2.display();

}

}

**5) Create class project(id,name,location).Define parameterized constructur.Keep a count of each object created nd display the details of each project**.

import scala.language.postfixOps

class project

{

var id=0

var name:String=null

var location:String=null

project.countobject()

def display()

{

println("Project Details are")

println("Id:"+id+"Name:"+name+"Loction:"+location)

}

def this(id:Int,name:String,location:String)

{

this()

this.id=id

this.name=name

this.location=location

}

}

object project

{

var count:Int=0

def countobject()={

count=count+1

}

}

object Class5

{

def main(args:Array[String])

{

var p1=new project(1,"App","Pune");

p1.display()

var p2=new project(2,"Website","Mumbai");

p2.display()

var p3=new project(3,"Game","Nashik");

p3.display()

var p4=new project(4,"WebApplication","Sangamner");

p4.display()

println("No of object cretead:"+project.count)

}

}

**6)Define a class sports(id,name,description,amount).Derive two classes Indoor and Outdoor.Define Appropriate constructors and operations.Create an object and perform operations.**

class Sport(protected var id:Int,protected var name:String,protected var description:String,protected var amount:Double){

}

class Indoor(Iid:Int,Iname:String,Idesc:String,Iamnt:Double) extends Sport(Iid,Iname,Idesc,Iamnt){

def info=(s"[Id:$id || name:$name || description:$description || amount:$amount]");

}

class Outdoor(Oid:Int,Oname:String,Odesc:String,Oamnt:Double) extends Sport(Oid,Oname,Odesc,Oamnt){

def info=(s"[Id:$id || name:$name || description:$description || amount:$amount]");

}

object SportOps{

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

var i1=new Indoor(5515,"Carrom","Board game",55885.32);

var i2=new Outdoor(55451,"Cricket","Bat ans Ball",68974.23);

println(i1.info);

println(i2.info);

}

}

**7) Design an abstract class Employee with computeSal() as an abstract function. Create two subclasses Worker and Manager. Salary of worker should be calculated on hourly basis of work and Salary of Manager should be calculated on monthly basis with additional incentives.**

abstract class Employee

{

var id:Int=0;

var name:String=null;

var salary:Double=0.0;

def computeSal();

}

class Worker extends Employee

{

def Accept()

{

println("\n enter the ID and name of worker");

id=scala.io.StdIn.readInt();

name=scala.io.StdIn.readLine();

}

def computeSal()

{

println("Enter no of hours work");

var hours=scala.io.StdIn.readDouble();

salary=hours\*100;

}

def Display()

{

println("\ndetails of worker are")

println("\tid="+id+ "\tname="+name+ "\tsalary="+salary);

}

}

class Manager extends Employee

{

def Accept()

{

println("enter the ID and name of manager");

id=scala.io.StdIn.readInt();

name=scala.io.StdIn.readLine();

}

def computeSal()

{

println("Enter salary of manager");

salary=scala.io.StdIn.readDouble();

println("Enter incentive bonus of manager");

var incentive=scala.io.StdIn.readDouble();

salary=salary+incentive;

}

def Display()

{

println("details of manager are")

println("\tid="+id+ "\tname="+name+ "\tsalary="+salary);

}

}

object Programe7

{

def main(args: Array[String])

{

var w=new Array[Worker](2);

var m=new Array[Manager](2);

for(i <- 0 to 1)

{

w(i)=new Worker();

w(i).Accept();

w(i).computeSal();

m(i)=new Manager();

m(i).Accept();

m(i).computeSal();

}

for(i <- 0 to 1)

{

w(i).Display();

m(i).Display();

}

}

}