**control structure**

1.write a program to calculate average of all no between n1&n2

->import scala.io.StdIn.\_;

object avgno

{

def main(args:Array[String])

{

println("Enter the number1");

var n=readInt()

println("Enter the number2");

var m=readInt()

var i=n;

var c=0;

var sum=0;

var ans=0

while(i<=m)

{

c=c+1;

sum=sum+i;

i=i+1;

}

ans=sum/c;

println("avg of "+n+"to"+m+"is"+ans);

}

}

Output:

Enter the number1

10

Enter the number2

20

avg of10to20is15

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

->import scala.io.StdIn.\_;

object fact

{

def main(args:Array[String])

{

println("Enter the number");

var n=readInt()

var f=1

while(n>0)

{

f=f\*n;

n-=1;

}

println("factorial of no:"+f)

}

}

Output:

Enter the number

5

factorial of given no:120

3.write a program to read five random number and check that random no are perfect no or not.

->import scala.io.StdIn.\_

import scala.util.control.\_

object perfect

{

def main(args:Array[String])

{

var r=scala.util.Random

var j=1

while(j<=5)

{

var n=r.nextInt(20)

println("random no is"+n)

var i=1

var sum=0

while(i<n)

{

if(n%i==0)

{

sum=sum+i

}

i=i+1

}

if(sum==n)

println(""+n+"no is perfect")

else

println(""+n+"no is not perfect")

j=j+1

}

}

}

Output:

random no is10

10no is not perfect

random no is5

5no is not perfect

random no is3

3no is not perfect

random no is4

4no is not perfect

random no is18

18no is not perfect

4.write a program to find second maximum number from given four numbers

->import scala.io.stdIn.\_;

object secondmax

{

def main(args:Array[String])

{

println("Enter the number1");

var a=readInt()

println("Enter the number2");

var b=readInt()

println("Enter the number3");

var c=readInt()

println("Enter the number4");

var d=readInt()

var n=0

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

n=a

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

n=b

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

n=c

else

n=d

if(n==a)

{

if(b<a&&(c<b && a<b))

println("second max b")

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

println("second max c")

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

println("second max d")

else

println("second largest a")

}

else if(n==b)

{

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

println("second max a")

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

println("second max c")

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

println("second max d")

else

println("second largest b")

}

else if(n==c)

{

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

println("second max a")

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

println("second max b")

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

println("second max d")

else

println("second largest c")

}

else if(n==d)

{

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

println("second max a")

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

println("second max b")

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

println("second max c")

else

println("second largest d")

}

}

}

Output:

Enter the number1

2

Enter the number2

4

Enter the number3

6

Enter the number4

1

second max b

5.write a program to calculate sum of prime number from 1to 100

->import scala.io.stdIn.\_

import scala.util.control.\_

object prime

{

def main(args:Array[String])

{

var n=4

var sum=0

var loop=new Breaks;

while(n<=100)

{

var i=2

var f=0

while(i<n)

{

loop.breakable{

if(n%i==0)

{

f=1;

loop.break;

}

}

i+=1

}

if(f==0)

sum=sum+n

n+=1

}

println("sum of 1 to 100 is"+sum)

}

}

Output:

sum of 1 to 100 is 1055

**ARRAY**

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

->import scala.io.StdIn.\_

import scala.util.control.\_

object maxarray

{

def main(args:Array[String])

{

var a=Array(20,10,30,15,7,90,50)

var max=0

var min=a(0)

for(n<-a)

{

if(max<n)

{

max=n

}

if(min>n)

{

min=n

}

}

println("Max no in array is"+max)

println("Min no in array is"+min)

}

}

output:-

C:\Users\SAI\scala\_samples\sbt\_proj>scalac maxarray.scala

C:\Users\SAI\scala\_samples\sbt\_proj>scala maxarray

Max no in array is90

Min no in array is7

2. Write a program to calculate transpose of a matrix.

->import scala.io.StdIn.\_

import scala.util.control.\_

object transpose

{

def main(args:Array[String])

{

var a=Array.ofDim[Int](3,3)

for(i<-0 to 2)

{

for(j<-0 to 2)

{

a(i)(j)=readInt()

}

}

for(i<-0 to 2)

{

for(j<-0 to 2)

{

print(" "+a(j)(i))

}

println(" ")

}

}

}

output:-

C:\Users\SAI\scala\_samples\sbt\_proj>scalac transpose.scala

C:\Users\SAI\scala\_samples\sbt\_proj>scala transpose

1 2 3

4 5 6

7 8 9

1 4 7

2 5 8

3 6 9

3. Write a program to calculate determinant of a matrix

->import scala.io.StdIn.\_

object determinant

{

def main(args:Array[String])

{

var a=Array.ofDim[Int](3,3)

var det=1

println("Enter matrix element")

for(i<-0 to 2)

{

for(j<-0 to 2)

{

a(i)(j)=readInt()

}

}

println("matrix is:")

for(i<-0 to 2)

{

for(j<-0 to 2)

{

print(" "+a(i)(j))

}

println(" ")

}

var x=(a(1)(1)\*a(2)(2))-(a(2)(1)\*a(1)(2))

var y=(a(1)(0)\*a(2)(2))-(a(2)(0)\*a(1)(2))

var z=(a(1)(0)\*a(2)(1))-(a(2)(0)\*a(1)(1))

for(i<-0 to 2)

{

for(j<-0 to 2)

{

det=(a(0)(0)\*x)-(a(0)(1)\*y)-(a(0)(2)\*z)

}

}

println("Determinant of matrix is::"+det)

}

}

output:-

C:\Users\SAI\scala\_samples\sbt\_proj>scalac determinant.scala

C:\Users\SAI\scala\_samples\sbt\_proj>scala determinant

Enter matrix element

1 2 3

4 5 6

7 8 9

matrix is:

1 2 3

4 5 6

7 8 9

Determinant of matrix is::18

4. Write a program to check if the matrix is upper triangular or not.

->import scala.io.StdIn.\_

import scala.util.control.\_

object upper

{

def main(args:Array[String])

{

var a=Array.ofDim[Int](3,3)

for(i<-0 to 2)

{

for(j<-0 to 2)

{

a(i)(j)=readInt()

}

}

var f1=0

var f2=0

val l1=new Breaks;

val l2=new Breaks;

for(i<-0 to 2)

{

for(j<-0 to 2)

{

if(i>j)

{

l1.breakable

{

if(a(i)(j)==0)

{

f1=1

l1.break

}

}

}

}

}

for(i<-0 to 2)

{

for(j<-0 to 2)

{

if(i<j)

{

l2.breakable

{

if(a(i)(j)!=0)

{

f2=1

l2.break

}

}

}

}

}

if(f1==0 && f2==0)

println("Uppertriangular matrix")

else

println("not Uppertriangular matrix")

}

}

output:-

C:\Users\SAI\scala\_samples\sbt\_proj>scalac upper.scala

C:\Users\SAI\scala\_samples\sbt\_proj>scala upper

1 0 0

2 3 0

4 5 6

Uppertriangular matrix

5. Write a program to sort the matrix using insertion sort.

->import scala.io.StdIn.\_

import scala.util.control.\_

object sort

{

def main(args:Array[String])

{

var a=Array.ofDim[Int](3,3)

var k=0

var key=0

for(i<-0 to 2)

{

for(j<-0 to 2)

{

a(i)(j)=readInt()

}

}

println("original matrix is:")

for(i<-0 to 2)

{

for(j<-0 to 2)

{

print(" "+a(i)(j))

}

println(" ")

}

var j=1

for(i<-0 to 2)

{

for(j<-0 to 2)

{

key=a(i)(j)

k=j-1

while(k>=0 && a(i)(k)>key)

{

a(i)(k+1)=a(i)(k)

k=k-1

}

a(i)(k+1)=key

}

a(i)(k+1)=key

}

println("After sorting matrix is:")

for(i<-0 to 2)

{

for(j<-0 to 2)

{

print(" "+a(i)(j))

}

println(" ")

}

}

}

output:-

C:\Users\SAI\scala\_samples\sbt\_proj>scalac sort.scala

C:\Users\SAI\scala\_samples\sbt\_proj>scala sort

1 3 2

5 4 6

6 7 8

original matrix is:

1 3 2

5 4 6

7 8 9

After sorting matrix is:

1 2 3

4 5 6

7 8 9

6. Write a program for multiplication of two matrices(Validate number of rows and columns before multiplication and give appropriate message)

->import scala.io.StdIn.\_

import scala.util.control.\_

object mul

{

def main(args:Array[String])

{

var a=Array.ofDim[Int](3,3)

var b=Array.ofDim[Int](3,3)

var c=Array.ofDim[Int](3,3)

println("Enter the first matrix elements:")

for(i<-0 to 2)

{

for(j<-0 to 2)

{

a(i)(j)=readInt()

}

}

println("Enter the second matrix elements:")

for(i<-0 to 2)

{

for(j<-0 to 2)

{

b(i)(j)=readInt()

}

}

println("matrix multiplication is:")

for(i<-0 to 2)

{

for(j<-0 to 2)

{

c(i)(j)=0

for(k<-0 to 2)

{

c(i)(j)+=a(i)(k)\*b(k)(j)

}

print( c(i)(j)+" ")

}

println(" ")

}

}

}

output:-

C:\Users\SAI\scala\_samples\sbt\_proj>scalac mul.scala

C:\Users\SAI\scala\_samples\sbt\_proj>scala mul

Enter the first matrix elements:

1 2 3

4 5 6

7 8 9

Enter the second matrix elements:

1 2 3

4 5 6

7 8 9

matrix multiplication is:

30 36 42

66 81 96

102 126 150

String

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

->import scala.io.StdIn.\_

object stringdemo2

{

def main(args:Array[String])

{

println("Enter the first string")

var s=readLine()

var l=s.length()

val a=Array.ofDim[Char](7)

var c=0

var j=0

for(i<-0 to l-1)

{

if(s.charAt(i)>=65 && s.charAt(i)<=91)

{

c=c+1

a(j)=s.charAt(i)

j=j+1

}

}

var l1=a.length()

for(j<-0 to l1-1)

println(" "+a(j))

var s1=a.mkString(" ")

println("lower case string"+s1.toLowerCase())

println("counter is"+c)

}

}

Output:

Enter the first string

RuPali

R

P

lower case string r p

counter is2

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

->import scala.io.StdIn.\_

object occString

{

def main(args:Array[String])

{

println("Enter the 1st string")

var s=readLine()

println("Enter the char")

var c=readChar()

var n=s.count(\_==c)

println("Total no of char is"+n)

}

}

output:-

C:\Users\SAI\scala\_samples\sbt\_proj>scalac occString.scala

C:\Users\SAI\scala\_samples\sbt\_proj>scala occString

Enter the 1st string

ooohh

Enter the char

o

Total no of char is3

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

->import scala.io.StdIn.\_

import java.lang.String;

object stringRemove

{

def main(args:Array[String])

{

println("Enter the first string")

var s1=readLine()

println("Enter the second string")

var s2=readLine()

var s3=s1.replaceAll(s2," ")

println("first string"+s1)

println("second string"+s2)

println("after replacement string"+s3)

}

}

output:-

C:\Users\SAI\scala\_samples\sbt\_proj>scalac stringRemove.scala

C:\Users\SAI\scala\_samples\sbt\_proj>scala stringRemove

Enter the first string

om

Enter the second string

sai

first stringom

second stringsai

after replacement stringom

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

->import scala.io.StdIn.\_

import java.lang.String;

object StringArray

{

def main(args:Array[String])

{

var a=Array.ofDim[String](10)

println("Enter the Array Limit")

var n=readInt()

for(i<-0 to n-1)

{

println("Enter the string")

a(i)=readLine()

}

for(i<-0 to n-1)

{

println(" "+a(i))

}

println("Enter the string for search & count occurrences")

var str1=readLine()

var c=0

for(i<-0 to n-1)

{

if(a(i).equals(str1))

c=c+1

}

println("Total no of string is"+c)

}

}

output:-

C:\Users\SAI\scala\_samples\sbt\_proj>scalac StringArray.scala

C:\Users\SAI\scala\_samples\sbt\_proj>scala StringArray

Enter the Array Limit

3

Enter the string

aaa

Enter the string

bbb

Enter the string

aaa

aaa

bbb

aaa

Enter the string for search & count occurrences

aaa

Total no of string is2

CLASSES AND OBJECT

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

->import scala.io.StdIn.\_

class currentAccount(val ano:Int,val name:String,val bal:Int,val minbal:Int)

{

var accno:Int=ano

var n:String=name

var balance:Int=bal

var mb:Int=minbal

def withdrawl()

{

println("Enter the withdrawl ammount:")

var amt=readInt()

if(amt<balance && mb<balance)

{

balance=balance-amt

}

else

{

println("Insufficient funds")

}

}

def deposite()

{

println("Enter the deposite amount")

var amt=readInt()

balance=balance+amt

}

def viewbalance()

{

println("Your balance is"+balance)

}

}

object amount

{

def main(args:Array[String])

{

var obj=new currentAccount(123,"priya",1200,500)

obj.withdrawl()

obj.viewbalance()

obj.deposite()

obj.viewbalance()

}

}

Output:

Enter the withdrawl ammount:

500

Your balance is700

Enter the deposite amount

200

Your balance is900

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

->import scala.io.StdIn.\_

class Employee

{

var id=0

var name=" "

var sal=0

def accept()

{

println("Enter the id,name,salary")

id=readInt()

name=readLine()

sal=readInt()

}

def display()

{

println(+id+" "+name+" "+sal)

}

}

object empclass

{

def main(args:Array[String])

{

println("Enter the no of employee")

var n=readInt()

var i=0

var obj:Array[Employee]=new Array[Employee](n)

while(i<n)

{

obj(i)=new Employee()

i=i+1

}

for(i<-0 to n-1)

{

obj(i).accept()

}

var max=obj(0).sal

for(i<-0 to n-1)

{

if(obj(i).sal>max)

max=obj(i).sal

}

for(i<-0 to n-1)

{

if(obj(i).sal==max)

obj(i).display()

}

}

}

Output:

Enter the no of employee

2

Enter the id,name,salary

1

rups

25000

Enter the id,name,salary

2

sid

30000

2 sid 30000

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

->import scala.io.StdIn.\_

abstract class Order1(val id:Int,val des:String)

{

var id1:Int=id

var des1:String=des

def accept()

def display()

}

class salesorder(val vendor:String,val cust:String) extends Order1(1,"aaa")

{

var v:String=vendor

var c:String=cust

def accept()

{

println("Enter the vendor name:")

v=readLine()

println("Enter the customer name:")

c=readLine()

}

def display()

{

println("\*\*\*\*\*\*\*\*\*\*\*Details\*\*\*\*\*\*\*\*\*")

println("Id="+id)

println("Des="+des)

println("Vendor name="+v)

println("Customer name="+c)

}

}

class purchesorder(val vendor:String,val cust:String) extends Order1(2,"bbb")

{

var v:String=vendor

var c:String=cust

def accept()

{

println("Enter the vendor name:")

v=readLine()

println("Enter the customer name:")

c=readLine()

}

def display()

{

println("\*\*\*\*\*\*\*\*\*\*\*Details\*\*\*\*\*\*\*\*\*")

println("Id="+id)

println("Des="+des)

println("Vendor name="+v)

println("Customer name="+c)

}

}

object order

{

def main(args:Array[String])

{

var ob1=new salesorder("","")

ob1.accept()

ob1.display()

var ob2=new purchesorder("","")

ob2.accept()

ob2.display()

}

}

Output:

Enter the vendor name:

kanchan

Enter the customer name:

aa

\*\*\*\*\*\*\*\*\*\*\*Details\*\*\*\*\*\*\*\*\*

Id=1

Des=aaa

Vendor name=kanchan

Customer name=aa

Enter the vendor name:

rr

Enter the customer name:

sai

\*\*\*\*\*\*\*\*\*\*\*Details\*\*\*\*\*\*\*\*\*

Id=2

Des=bbb

Vendor name=rr

Customer name=sai

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

->import scala.io.StdIn.\_

abstract class shape1

{

def volume()

def display()

var a=0.0

var h=0.0

var r=0.0

var v=0.0

}

class cube extends shape1

{

def volume()

{

println("Enter the value of a=")

a=readFloat()

v=a\*a\*a

}

def display()

{

println("volume of cube="+v)

}

}

class cylinder extends shape1

{

def volume()

{

println("Enter the radius and height=")

r=readFloat()

h=readFloat()

v=3.14\*r\*r\*h

}

def display()

{

println("volume of cylinder="+v)

}

}

object shape

{

def main(args:Array[String])

{

var ob1=new cube()

ob1.volume()

ob1.display()

var ob2=new cylinder()

ob2.volume()

ob2.display()

}

}

Output:

Enter the value of a=

3

volume of cube=27.0

Enter the radius and height=

2

3

volume of cylinder=37.68

LIST

1.Create Lists using five different methods( Lisp style , Java style, fill, range and tabulate methods)

->import java.io.\_

object ListDemo

{

def main(args: Array[String])

{

println("Create a Scala List in the Lisp style")

val list = 1 :: 2 :: 3 :: Nil

println(list)

println("Create a Scala List in the Java style")

val list1 = List(1,2,3)

println(list1)

val x = List[Number](1, 2.0, 33d, 0x1)

println(x)

println("Create a Scala List with the range method")

val x1 = List.range(1, 10)

println(x1)

val x2 = List.range(0, 10, 2)

println(x2)

println("Create a Scala List with the List class fill method")

val x3 = List.fill(3)("foo")

println(x3)

println("Create a Scala List with the List class tabulate method")

val x4 = List.tabulate(5)(n => n \* n)

println(x4)

}

}

Output:

Create a Scala List in the Lisp style

List(1, 2, 3)

Create a Scala List in the Java style

List(1, 2, 3)

List(1, 2.0, 33.0, 1)

Create a Scala List with the range method

List(1, 2, 3, 4, 5, 6, 7, 8, 9)

List(0, 2, 4, 6, 8)

Create a Scala List with the List class fill method

List(foo, foo, foo)

Create a Scala List with the List class tabulate method

List(0, 1, 4, 9, 16)

2.create a two list and merge it and store the sorted in ascending order.

->import scala.collection.mutable.ListBuffer

object List2

{

def main(args:Array[String])

{

var x=Seq(2,5,3,6,7)

var y=Seq(1,8,4,77,11)

var z=new ListBuffer[Int]()

var i=0

var j=0

var k=0

var n=x.length

var m=y.length

x=x.sorted

y=y.sorted

while(i<n-1 && j<m-1)

{

if(x(i)<y(j))

{

z+=x(i)

i=i+1

}

if(x(i)>y(j))

{

z+=y(j)

j=j+1

}

}

while(i<=n-1)

{

z+=x(i)

i=i+1

}

while(j<=m-1)

{

z+=y(j)

j=j+1

}

println(" "+x)

println(" "+y)

println(" "+z.toList)

}

}

Output:

List(2, 3, 5, 6, 7)

List(1, 4, 8, 11, 77)

List(1, 2, 3, 4, 5, 6, 7, 8, 11, 77)

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

->object List3

{

def main(args:Array[String])

{

var x=List.range(1,51)

var i=0

while(i<50)

{

if(x(i)%3==0)

println(" "+x(i))

i=i+1

}

}

}

Output:

3

6

9

12

15

18

21

24

27

30

33

36

39

42

45

48

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

->import scala.collection.mutable.ListBuffer

object list4

{

def main(args:Array[String])

{

var x=List.range(1,11)

var z=new ListBuffer[Int]()

var i=0

var j=0

while(i<10)

{

if(x(i)%2==0)

{

z+=x(i)

}

i=i+1

}

println(" "+z.toList)

var mul=1

while(j<1)

{

mul=mul\*z(j)

j=j+1

}

println(" "+mul)

}

}

Output:-

List(2, 4, 6, 8, 10)

2

5.Write a program to create list with 10 members using function 3n2+4n+6

->import java.io.\_

import scala.collection.mutable.ListBuffer

object List5

{

def main(args: Array[String])

{

var x=List(1)

for (i <- 1 to 10)

{

x=x:::Accept(i)::Nil

}

println(x)

}

def Accept(n:Int) : Int =

{

println("Round : "+n)

return ((3\*n\*n)+(4\*n)+6)

}

}

Output:

Round : 1

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

->import java.io.\_

import scala.collection.mutable.ListBuffer

object List10

{

def main(args: Array[String])

{

var x=List(1)

for (i <- 1 to 100)

{

x=x:::(i)::Nil

}

println(x)

var x1=List(1)

for (i <- 1 to 101)

{

if(i%10==0)

x1=x1:::(i)::Nil

}

println(x1)

}

}

Output:

List(1, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100)

List(1, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100)

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

->import java.io.\_

import scala.collection.mutable.ListBuffer

object List50

{

def main(args: Array[String])

{

var x=List(1)

var x1=List(1)

for (i <- 1 to 50)

{

x=x:::Accept(i)::Nil

if(Accept(i)%7==0)

{

}

else

{

x1=x1:::Accept(i)::Nil

}

}

println(x)

println("\n\n\n\n\n\n\n\n")

println(x1)

}

def Accept(n:Int) : Int =

{

return ((2\*n)+3)

}

}

Output:

List(1, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65, 67, 69, 71, 73, 75, 77, 79, 81, 83, 85, 87, 89, 91, 93, 95, 97, 99, 101, 103)

List(1, 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, 103)

Round : 2

Round : 3

Round : 4

Round : 5

Round : 6

Round : 7

Round : 8

Round : 9

Round : 10

List(1, 13, 26, 45, 70, 101, 138, 181, 230, 285, 346)

**Map**

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

->object map1

{

def main(args:Array[String])

{

val m1: Map[Int, String] = Map(1->"Rahul",2->"sai",3->"sam",4->"om");

fun(m1)

}

def fun(m1:Map[Int,String])

{

m1.keys.foreach

{

i =>

println(" "+m1(i).toUpperCase())

}

}

}

output:-

C:\Users\SAI\scala\_samples\sbt\_proj>scalac map1.scala

C:\Users\SAI\scala\_samples\sbt\_proj>scala map1

RAHUL

SAI

SAM

OM

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

->object map2

{

def main(args:Array[String])

{

val m1: Map[Int,String] = Map(1->"mayuri",2->"ashvini",3->"siddhi");

fun(m1)

}

def fun(m1:Map[Int,String])

{

val find=m1(1)

m1.keys.foreach

{

i =>

if(find.equals(m1(i)))

{

println(" "+m1(i)+" "+i)

}

}

}

}

output:-

C:\Users\SAI\scala\_samples\sbt\_proj>scalac map2.scala

C:\Users\SAI\scala\_samples\sbt\_proj>scala map2

mayuri 1

**Set**

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

->object set1

{

def main(args:Array[String])

{

val s1=Set(1,2,3,4,5);

val s2=Set(1,3,4,7,8);

println("Common elements in set are:"+s1.intersect(s2))

}

}

output:-

C:\Users\SAI\scala\_samples\sbt\_proj>scalac set1.scala

C:\Users\SAI\scala\_samples\sbt\_proj>scala set1

Common elements in set are:HashSet(1, 3, 4)

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

->object set2

{

def main(args:Array[String])

{

val s1=Set(5,6,9,20,30,45);

println( "largest element " + s1.max )

println( " smallest element " + s1.min )

}

}

output:-

C:\Users\SAI\scala\_samples\sbt\_proj>scalac set2.scala

C:\Users\SAI\scala\_samples\sbt\_proj>scala set2

largest element 45

smallest element 5

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

->import scala.collection.immutable.\_

import scala.collection.immutable.HashSet

import scala.math.Numeric

object set3

{

def main(args:Array[String])

{

val s1=Set(1,2,3,4)

val s2=Set(5,6,7,8)

val s=s1 union s2

println(" "+s)

val n=s.sum

val m=n/s.size

println(" Sum is:"+n)

println(" Avg is:"+m)

}

}

output:-

C:\Users\SAI\scala\_samples\sbt\_proj>scalac set3.scala

C:\Users\SAI\scala\_samples\sbt\_proj>scala set3

HashSet(5, 1, 6, 2, 7, 3, 8, 4)

Sum is:36

Avg is:4