**[ 2CEIT5PE5: MOBILE APPLICATION DEVELOPMENT]**

Practical: 1



**AIM- KOTLIN PROGRAMS**

Submitted By:

21012021003

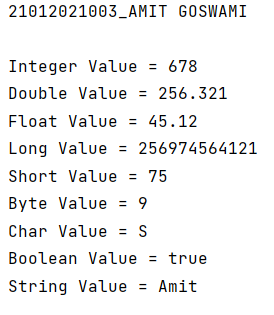


**Department of Computer Engineering/Information Technology**

1. **Store & Display values in different variable of different type (Integer, Double, Float, Long, Short, Byte, Char, Boolean, String).**

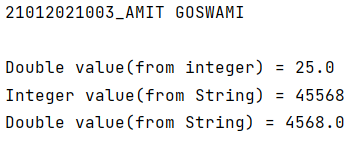
fun main()  
{  
 println("21012021003\_AMIT GOSWAMI\n")  
  
 val a : Int = 678  
 val b : Double = 256.321  
 val c : Float = 45.12f  
 val d : Long = 256974564121  
 val e : Short = 75  
 val f : Byte = 9  
 val g : Char = 'S'  
 val h : Boolean = true  
 val i : String = "Amit"  
  
 println("Integer Value = $a")  
 println("Double Value = $b")  
 println("Float Value = $c")  
 println("Long Value = $d")  
 println("Short Value = $e")  
 println("Byte Value = $f")  
 println("Char Value = $g")  
 println("Boolean Value = $h")  
 println("String Value = $i")  
}

}



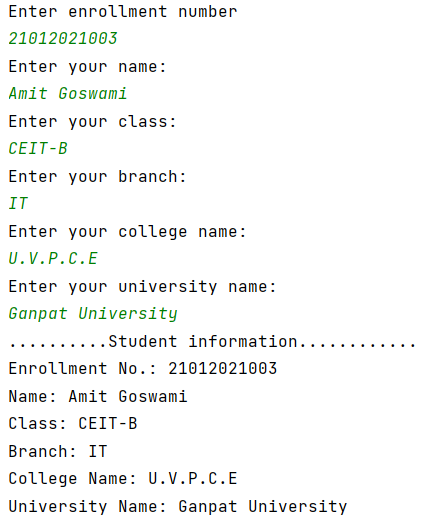
1. **Type conversion:Integer to Double, String to Integer, String to Double.**

fun main()  
{  
 println("21012021003\_AMIT GOSWAMI\n")  
  
 val a1 : Int = 25  
 val a2=a1.toDouble()  
 println("Double value(from integer) = $a2")  
  
 val b1 : String = "45568"  
 val b2=b1.toInt()  
 println("Integer value(from String) = $b2")  
  
 val c1 : String = "4568"  
 val c2 = c1.toDouble()  
 println("Double value(from String) = $c2")  
}



1. **Scan student’s information and display all the data.**

fun main(){  
 println("Enter enrollment number");  
 var a = readLine();  
  
 println("Enter your name: ");  
 var b = readLine();  
  
 println("Enter your class: ");  
 var c = readLine();  
  
 println("Enter your branch: ");  
 var d = readLine();  
  
 println("Enter your college name: ");  
 var e = readLine();  
  
 println("Enter your university name: ");  
 var f = readLine();  
  
  
 println("..........Student information............");  
  
 println("Enrollment No.: $a");  
 println("Name: $b");  
 println("Class: $c");  
 println("Branch: $d");  
 println("College Name: $e");  
 println("University Name: $f");  
  
}



1. **Find the number is odd or even by using Control Flow inside println() method.**

import java.util.Scanner

fun main() {

var reader = Scanner(System.`in`)

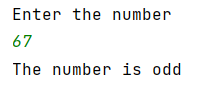
println("enter the number : ")

var x:Int = reader.nextInt()

var ans:String = if(x.toInt() %2 == 0) "even" else "odd"

println("$x is $ans")

}



1. **Display month name using When.**

fun main()

{

print("Enter month number : ")

when (readLine()!!.toInt()) {

1 -> print("MONTH = January")

2 -> print("MONTH = February")

3 -> print("MONTH = March")

4 -> print("MONTH = April")

5 -> print("MONTH = May")

6 -> print("MONTH = June")

7 -> print("MONTH = July")

8 -> print("MONTH = August")

9 -> print("MONTH = September")

10 -> print("MONTH = October")

11-> print("MONTH = November")

12 -> print("MONTH = December")

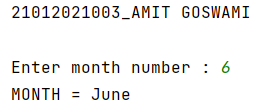
else -> {

print("Enter proper month number")

}

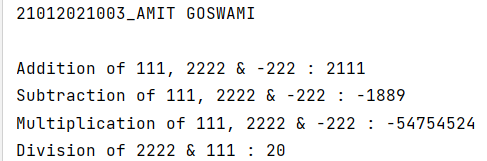
}

}



1. **By using a user defined function perform all arithmetic operations.**

fun main()  
{  
 println("21012021003\_AMIT GOSWAMI\n")  
  
 *cal*(111,2222,-222)  
}  
fun cal(a:Int, b:Int, c:Int)  
{  
 println("Addition of $a, $b & $c : ${a+b+c}")  
 println("Subtraction of $a, $b & $c : ${a-b-c}")  
 println("Multiplication of $a, $b & $c : ${a\*b\*c}")  
 println("Division of $b & $a : ${b/a}")  
}



1. **Find the factorial of number by recursion. Explain "tailrec" keyword.**

fun main()

{

print("ENTER NUMBER : ")

val n : Int = readLine()!!.toInt()

println("Factorial of $n : ${fact(n)}")

println("By Tailrec Keyword, Factorial of $n : ${factTail(n)}")

}

fun fact(n:Int):Int

{

var f=1

for (i in 1..n) {

f \*= i

}

return f

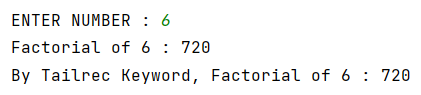
}

tailrec fun factTail(n:Int):Int

{

return if (n==1) 1 else n \* factTail(n-1)

}



1. **Create different types of Array as shown in image. Explore Arrays.deepToString(), contentDeepToString() methods, IntArray variable .joinToString() and use in program to print Array. Explore range, downTo, until etc. for loop and use in this program. Sort Array of Integer data type without using inbuilt function & with using inbuilt function.**

fun main(){

println("Array created by using arrayof() method")

var a1 = arrayOf(23,43,5,63,8)

println(a1.contentToString())

println("Array created by using Array<> method")

var a2 = Array(5){0}

println(a2.contentDeepToString())

println("Array created by using Array<> and Lambda function")

var a3 = Array(7){i->i}

println(a3.contentToString())

println("Array created using IntArray()")

var a4 = IntArray(5){7}

println(a4.joinToString (", "))

println("Array created using IntArrayOf()")

var a5 = intArrayOf(12,54,64,67,2)

println(a5.joinToString(", "))

println("Array created by using arrayof() and intArrayOf()")

var a6 = arrayOf(intArrayOf(45,3), intArrayOf(4,1), intArrayOf(9,2))

println(a6.contentDeepToString())

print("Enter number of Elements : ")

val size : Int = readLine()!!.toInt()

val user = IntArray(size) {0}

for(i in 0 until size)

{

print("a[$i]:")

user[i] = readLine()!!.toInt()

}

print("Entered Array = ")

println(user.contentToString())

println("========================== With In-Built Function ==========================")

println("Array Sorted by in-built Function = ")

user.sort()

println(user.contentToString())

val a7 = intArrayOf(45,96,689,593,-45,-76,-453,-56)

println("========================== Without In-Built Function ==========================")

println("Array Sorted without in-built Function = ")

println(user.contentToString())

var temp: Int

for (i in a7.indices) {

for (j in a7.indices) {

if (a7[j] > a7[i]) {

temp = a7[j]

a7[j] = a7[i]

a7[i] = temp

}

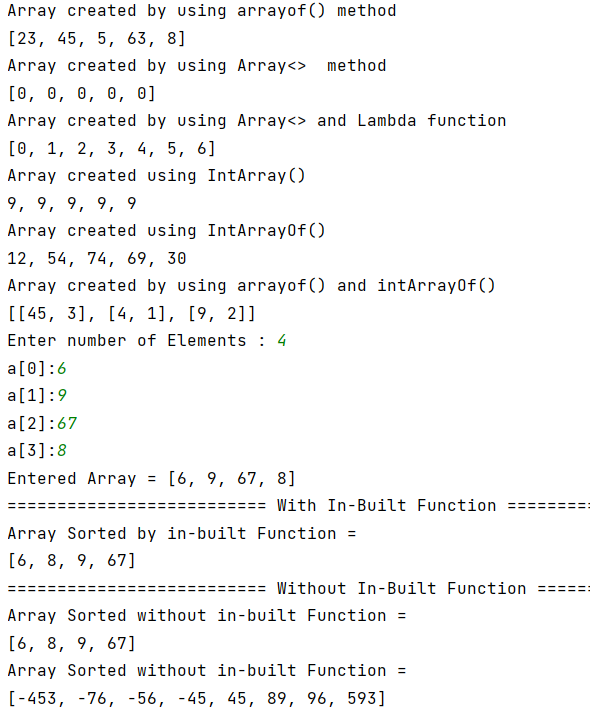
}

}

println("Array Sorted without in-built Function = ")

println(a7.contentToString())

}

****

1. **Find the max number from ArrayList.**

fun main()

{

print("Enter Elements : ")

val n:Int = readLine()!!.toInt()

val num = ArrayList<Int>()

var maxnum = 0

for (i:Int in 0 until n)

{

print("nums[$i] = ")

num.add(i,readLine()!!.toInt())

if (maxnum<num[i])

{

maxnum = num[i]

}

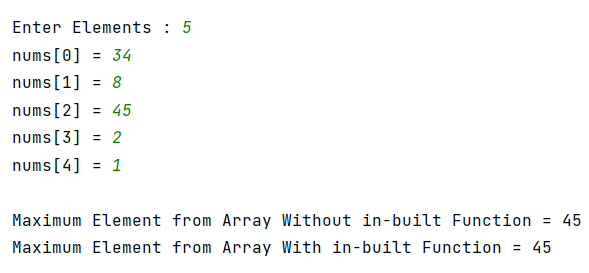
}

println()

println("Maximum Element from Array Without in-built Function = $maxnum")

println("Maximum Element from Array With in-built Function = ${num.max()}")

}



1. **Write Different types of Class & Constructor. Create a class Car and set various members like type, model, price, owner, milesDrive. add the function getCarPrice in it. Create an object of Car class and access property of it. (getCarInformation(), getOriginalCarPrice(), getCurrentCarPrice(), displayCarInfo() etc.)**

fun main()

{

val car1 = Car("MERCEDES, 2018", "RAM", 1250, 100000.0, 98950.0)

car1.getCarFullDetails()

val car2 = Car("BMW, 2019", "KRISHNA", 200, 400000.0, 399800.0)

car2.getCarFullDetails()

val Cars = ArrayList<Car> (2)

val car3 = Car("KOENIGSEGG, 2017","KJS",1000,1000000.0,700000.0)

val car4 = Car("MAHINDRA, 2020","NPP",2000,4000000.0,3000000.0)

Cars.add(car3)

Cars.add(car4)

for (i in Cars)

{

println("-------------------")

i.getCarFullDetails()

}

}

class Car(private val model: String, private val owner: String, private val miles: Int, private val original: Double, private val current: Double)

{

init

{

println("Object of class is Created and Init is Called.")

}

private fun info(): String

{

return model

}

private fun carowner(): String

{

return owner

}

private fun milesDrive(): Int

{

return miles

}

private fun orgprice(): Double

{

return original

}

private fun currprice(): Double

{

return current

}

fun getCarFullDetails()

{

println("===================")

println("Car Information : ${info()}")

println("Car owner : ${carowner()}")

println("Miles Drive : ${milesDrive()}")

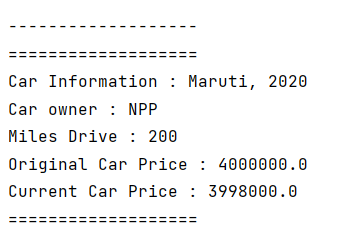
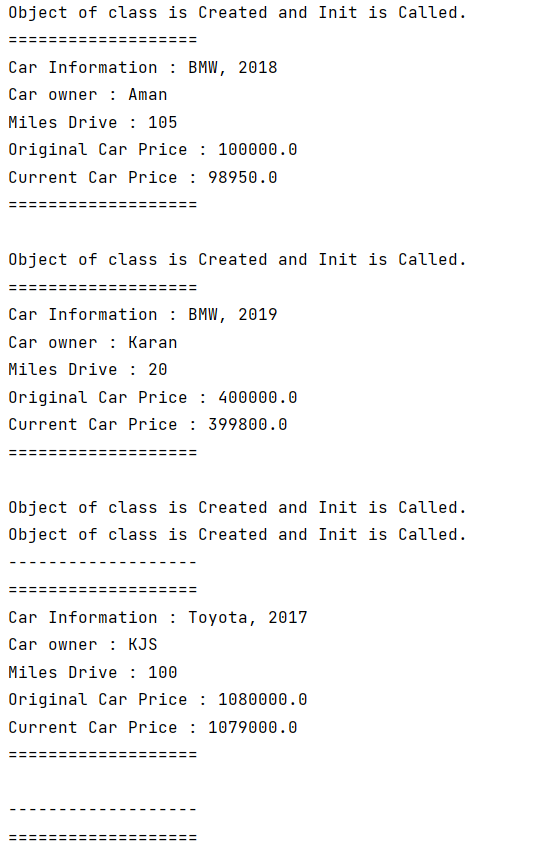
println("Original Car Price : ${orgprice()}")

println("Current Car Price : ${currprice()}")

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

}

}



1. **Write about Operator Overloading. Perform Matrix Addition, Subtraction & Multiplication using Class Matrix & operator overloading. Overload toString() function in Matrix class.**

fun main()

{

val firstMatrix = Matrix(arrayOf(intArrayOf(1, 2, 3), intArrayOf(4, 5, 6)), 2, 3)

val secondMatrix1 = Matrix(arrayOf(intArrayOf(1, 4), intArrayOf(2, 5), intArrayOf(3, 6)), 3, 2)

val secondMatrix2 = Matrix(arrayOf(intArrayOf(7, 4), intArrayOf(8, 5), intArrayOf(9, 6)), 3, 2)

println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Addition \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print("Matrix 1 : ")

println(secondMatrix2.toString())

print("Matrix 2 : ")

println(secondMatrix1.toString())

val addMatrix = secondMatrix2 + secondMatrix1

print("Addition : $addMatrix")

println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Subtraction \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print("Matrix 1 : ")

println(secondMatrix2.toString())

print("Matrix 2 : ")

println(secondMatrix1.toString())

val subMatrix = secondMatrix2 - secondMatrix1

print("Subtraction : $subMatrix")

println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Multiplication \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

print("Matrix 1 : ")

println(firstMatrix.toString())

print("Matrix 2 : ")

println(secondMatrix1.toString())

val mulMatrix = firstMatrix \* secondMatrix1

println("Multiplication : $mulMatrix")

}

class Matrix(private val matrix: Array<IntArray>, private val rows: Int, private val cols: Int)

{

override fun toString(): String

{

var res = "($rows x $cols Matrix): \n"

for (i in matrix)

{

for (j in i)

{

res += "$j\t"

}

res += "\n"

}

return res

}

operator fun plus(obj: Matrix): Matrix

{

val sum = Array(this.rows) { IntArray(this.cols) }

for (i in 0 until this.rows)

{

for (j in 0 until this.cols)

{

sum[i][j] = this.matrix[i][j] + obj.matrix[i][j]

}

}

return Matrix(sum, this.rows, this.cols)

}

operator fun minus(obj: Matrix): Matrix

{

val sub = Array(this.rows) { IntArray(this.cols) }

for (i in 0 until this.rows)

{

for (j in 0 until this.cols)

{

sub[i][j] = this.matrix[i][j] - obj.matrix[i][j]

}

}

return Matrix(sub, this.rows, this.cols)

}

operator fun times(obj: Matrix): Matrix

{

val mul = Array(this.rows) { IntArray(obj.cols) }

for (i in 0 until this.rows)

{

for (j in 0 until obj.cols)

{

mul[i][j] = 0

for (k in 0..obj.cols)

{

mul[i][j] += this.matrix[i][k] \* obj.matrix[k][j]

}

}

}

return Matrix(mul, this.rows, obj.cols)

}

}

