Practical 04 – 22001212

01)

22001212_q1.scala > {} InventoryManagement > main

```scala
object InventoryManagement {

  var itemNames: Array[String] = Array("Sugar", "Rice", "Dhal","Flour");
  var itemQuantities: Array[Int] = Array(20, 55, 30,45);

  def displayInventory(names: Array[String], quantities: Array[Int]): Unit = {
    println("Current Inventory:");
    println("Item Name"+" "*4+"|"+"Quantity");
    for ((name, quantity) <- names zip quantities) {
      println(f"$name%-12s | $quantity");
    }
    println();
  }

  def restockItem(names: Array[String], quantities: Array[Int], itemName: String, quantity: Int): Unit = {
    val i = names.indexOf(itemName);
    if(i != -1){
        quantities(i)+=quantity;
        println(quantity+" of "+itemName+" restocked to the system");
    }
    else{
        println("Item is not found to restock");
    }
  }

  def sellItem(names: Array[String], quantities: Array[Int], itemName: String, quantity: Int): Unit = {
    val i=names.indexOf(itemName);
    if(i != -1){
        if(quantities(i)>=quantity){
            quantities(i)-=quantity;
            println(quantity+" of "+itemName+" sold");
        }
```

22001212_q1.scala > {} InventoryManagement > main

```scala
object InventoryManagement {
  def sellItem(names: Array[String], quantities: Array[Int], itemName: String, quantity: Int): Unit = {
        else{
            println(itemName+"is not enough to sell");
        }
    }
    else{
        println("Item is not found to sell");
    }
    println();
  }

  def main(args: Array[String]): Unit = {
    displayInventory(itemNames, itemQuantities);
    restockItem(itemNames, itemQuantities, "Flour", 50);
    sellItem(itemNames, itemQuantities, "Sugar", 15);
    displayInventory(itemNames, itemQuantities);
  }
}
```

```
[Running] scala "c:\Users\User\Desktop\UCSC\2nd Year Sem
there were 2 deprecation warnings; re-run with -deprecat
1 warning found
Current Inventory:
Item Name     |Quantity
Sugar         | 20
Rice          | 55
Dhal          | 30
Flour         | 45

50 of Flour restocked to the system
15 of Sugar sold

Current Inventory:
Item Name     |Quantity
Sugar         | 5
Rice          | 55
Dhal          | 30
Flour         | 95


[Done] exited with code=0 in 9.704 seconds
```

02)

22001212_q2.scala > {} NumberPatternMatcher > inputMatcher

```scala
object NumberPatternMatcher{
    def inputMatcher(num:Int):Unit= num match{
        case x if(x<=0)=>println("Negative number/Zero is input");
        case x if(x%2==0)=>println("Even number is given");
        case _=>println("Odd number is given");
    }

    def main(args:Array[String]):Unit={
        print("Input Number>");
        val num=scala.io.StdIn.readInt();
        inputMatcher(num);
    }
}
```

```
PS C:\Users\User\Desktop\UCSC\2nd Year Sem-01\SCS 2204 - Functional Programming\Practical 04> scala 22001212_q2.scala
Input Number>88
Even number is given
PS C:\Users\User\Desktop\UCSC\2nd Year Sem-01\SCS 2204 - Functional Programming\Practical 04> scala 22001212_q2.scala
Input Number>-5
Negative number/Zero is input
PS C:\Users\User\Desktop\UCSC\2nd Year Sem-01\SCS 2204 - Functional Programming\Practical 04>
```

03)

22001212_q3.scala > {} StringFormat > main

```scala
object StringFormat{

    def toUpper(string:String):String={
        return string.toUpperCase();
    }

    def toLower(string:String):String={
        return string.toLowerCase();
    }

    def formatNames(name: String)(formatFunc: String => String): String = formatFunc(name);

    def main(args:Array[String]):Unit={
        val inputs=List("Benny","Niroshan","Saman","Kumara");
        for(i<- 0 to inputs.length-1){
            if(inputs(i)=="Benny"){
                println(toUpper(inputs(i)));
            }
            else if(inputs(i)=="Niroshan"){
                val(firstTwo,last)=inputs(i).splitAt(2);
                println(toUpper(firstTwo)+toLower(last));
            }
            else if(inputs(i)=="Saman"){
                println(toLower(inputs(i)));
            }
            else if(inputs(i)=="Kumara"){
                val t=inputs(i).last;
                val mid=inputs(i).tail.init;
                println(inputs(i).head+toLower(mid)+toUpper(t.toString()));
            }
        }
    }
}
```

```
[Running] scala "c:\Users\User\Desktop\UCSC\2nd Year Sem-01\SCS
there was 1 deprecation warning; re-run with -deprecation for d
1 warning found
BENNY
NIroshan
saman
KumarA

[Done] exited with code=0 in 7.833 seconds
```