29/02/2024

Name : P.NarendraKumar
ID : I07

Today Topics :-

1. **SCALA – TRAITS**

   1.Value classes and Universal traits
   2. When to Use Traits?

Code 1 : (Trait.scala)

```
trait sample
{
   def method1(x:Int):Int
   def method2(x:Int,y:Int):Int=x+y;
}
class child1 extends sample{
   def method3():Unit=
      {
         println("in class")
      }
   def method1(x:Int):Int=
      {
         x
      }
}

object trait1
{
   def main(args: Array[String]): Unit = {
      var obj=new child1()
      obj.method3();
      println(obj.method2(10,20));
      println("passing value is:"+obj.method1(1800))
   }
}
```

Code 2 : (Value classes.scala)

```
class s(val value:Double) extends AnyVal
{
   def method:Double = value*100;
}
object valueClass{
  def main(args: Array[String]): Unit = {
        val obj=new s(10)
        println(obj.method)
    }
}
```

Additional Program (BubbleSort)

```
object BubbleSort {
  def bubbleSort(arr: Array[Int]): Array[Int] = {
    val n = arr.length
    for (i <- 0 until n - 1) {
      for (j <- 0 until n - i - 1) {
        if (arr(j) > arr(j + 1)) {
          // Swap arr(j) and arr(j + 1)
          val temp = arr(j)
          arr(j) = arr(j + 1)
          arr(j + 1) = temp
        }
      }
    }
    arr
  }

  def main(args: Array[String]): Unit = {
    val arr = Array(64, 34, 25, 12, 22, 11, 90)
    println("Original array:")
    println(arr.mkString(", "))

    val sortedArr = bubbleSort(arr)
    println("\nSorted array:")
    println(sortedArr.mkString(", "))
  }
}
```

2.SCALA – PATTERN MATCHING

Code 3 : (PatternMatching.scala)

```
object pattern{
    def matchNumber(x: Int): String = x match {
        case 0 => "Zero"
        case 1 => "One"
        case 2 => "Two"
        case _ => "Other"
    }
```

```
  def main(args: Array[String]): Unit = {
    println(matchNumber(1))  // Output: One
    println(matchNumber(5))  // Output: Other
  }
}
```

Code 4 :

```
object Demo {
  def main(args: Array[String]) {
    println(matchTest("two"))
    println(matchTest("test"))
    println(matchTest(1))
  }

  def matchTest(x: Any): Any = x match {
    case 1 => "one"
    case "two" => 2
    case y: Int => "scala.Int"
    case _ => "many"
  }
}
```

Code 5 : (Matching using Case Classes)

```
object Demo {
  def main(args: Array[String]) {
    val alice = new Person("Alice", 25)
    val bob = new Person("Bob", 32)
    val charlie = new Person("Charlie", 32)

    for (person <- List(alice, bob, charlie)) {
      person match {
        case Person("Alice", 25) => println("Hi Alice!")
        case Person("Bob", 32) => println("Hi Bob!")
        case Person(name, age) => println(
          "Age: " + age + " year, name: " + name + "?")
      }
    }
  }
  case class Person(name: String, age: Int)
}
```