```scala
//Scala Tutorial 2
//20001258 - B.P.P.T.P.Pathirana

object Question1 {

  def increment(z: Int): Int = {
    return z + 1;
  }

  def main(args: Array[String]) {

    var k, i, j = 2;
    var m, n = 5;
    var f = 12.0f;
    var g = 4.0f;
    var c = 'X'

    println(k + 12 * m);
    println(m / j);
    println(n % j);
    println(m / j * j);
    println(f + 10 * 5 + g);
    println(increment(i) * n);

  }
}
```

Question 2

| Scala | Java |
|---|---|
| • Scala is a statically typed programming language. | • Java is a multi-platform, network-centric, programming language. |
| • Scala variables are by default immutable type. | • Java variables are by default mutable type. |
| • Scala doesn’t contain static members. | • Java contains static members. |
| • Scala supports operator overloading. | • Java doesn’t support operator overloading. |
| • Scala doesn’t offer backward compatibility. | • Java offers backward compatibility. |
| • Scala is less readable because of nested code. | • Java is more readable. |

```
object Question3 {

  def increment(z: Int): Int = {
    return z + 1;
  }

  def decrement(z: Int): Int = {
    return z - 1;
  }

  def main(args: Array[String]) {

    var a = 2;
    var b = 3;
    var c = 4;
    var d = 5;
    var k = 4.3f;
    var g = 4.0f;

    println(decrement(b) * a + c * d); //- -b * a + c *d - -
    b = b - 1;
    d = d - 1;

    println(a); //a++
    a = a + 1;

    println(-2 * (g - k) + c);

    println(c); //c++
    c = c + 1;

    println(increment(c) * a); //c = ++c * a++
    c = c + 1;
    a = a + 1;

    c = (c + 1) * a;

  }
}
```

```
object Question4a {

  def normal(hours: Int): Int = hours * 250;
  def ot(hours: Int): Int = hours * 85;
  def total(h1: Int, h2: Int): Int = {
    normal(h1) + ot(h2);
  }
  def tax(total: Int): Double = {
    total * 0.12;
  }

  def finalSalary(h1: Int, h2: Int): Double = {
    total(h1, h2) - tax(total(h1, h2));
  }

  def main(args: Array[String]) {

    println(finalSalary(40, 30));

  }
}

object Question4b  {
  def noOfatendees(ticketPrice: Int): Int = 120 + (15 - ticketPrice) / 5 * 20;
  def revenue(price: Int): Int = noOfatendees(price) * price;
  def cost(price: Int): Int = 500 + 3 * noOfatendees(price);
  def profit(price: Int): Int = revenue(price) - cost(price);

  def main(args: Array[String]) {
    println(profit(25), profit(30), profit(35));
     //profit maximize when ticket price becomes Rs.25
    println(profit(10), profit(15), profit(20));
  }
}
```

Git Hub Link: https://github.com/rusha-99/Scala-Tutorial-2.git