```scala
import collection.mutable.HashMap

object Change {
  //val currency = List(25, 10, 5, 1)
  val currency = List(240,60,30,24,12,6,3,1)

  def getDynamicChange(amt: Int) = {
    // Get the change and put it in an array using a dynamic
    // algorithm.
    val data = new Array[Float](amt + 1)
    val denom = new Array[Float](amt + 1)
    data(0) = 0
    for (j <- 1 until amt + 1) {
      data(j) = Float.PositiveInfinity
      for (i <- currency) {
        if (j >= i && 1 + data(j - i) < data(j)) {
          data(j) = 1 + data(j - i)
          denom(j) = i
        }
      }
    }

    // Loop through and get the letters.
    val finalLetters = new HashMap[Int, Int]
    var j = amt
    while (j > 0) {
      val x = denom(j).toInt
      if (finalLetters.contains(x)) {
        finalLetters(x) += 1
      } else {
        finalLetters(x) = 1
      }
      j -= x
    }

    // Get a string of the results
    var out: List[String] = Nil
    for ((denom, cnt) <- finalLetters) {
      out ::= "%d(%d)".format(cnt, denom)
    }

    out.reverse.reduceLeft(_ + " + " + _)
  }

  def getChange(amt: Int) = {
    var out: List[String] = Nil
    var workingAmt = amt
    for (x <- currency) {
      if (workingAmt / x > 0) {
        out ::= "%d(%d)".format(workingAmt / x, x)
      }
      workingAmt = workingAmt % x
    }

    out.reverse.reduceLeft(_ + " + " + _)
  }

  def usage = {
    println("USAGE: change AMOUNT")
    System.exit(1)
  }

  def main(args: Array[String]) = {
    if (args.length != 1) usage
```

```
    // Call the algorithm that will make change for a regular currency.
    val amt = args(0).toInt
    println("Greedy algorithm: " + getChange(amt))
    println("Dynamic Programming algorithm: " + getDynamicChange(amt))
  }
}
```