Basics of Scala

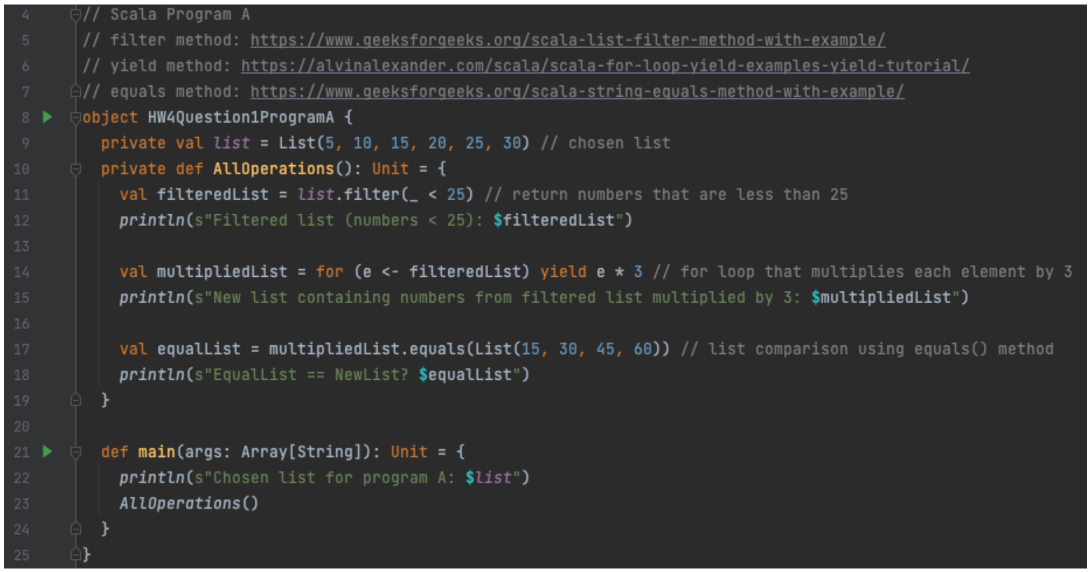
Vicente De Leon

UID: 2001014594

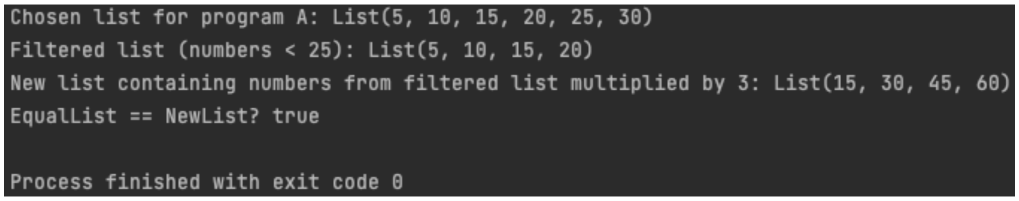
Assignment 4

1. Question 1
   Program A
   Resources:
   - Filter method: https://www.geeksforgeeks.org/scala-list-filter-method-with-example/
   - Yield method: https://alvinalexander.com/scala/scala-for-loop-yield-examples-yield-tutorial/
   - Equal method: https://www.geeksforgeeks.org/scala-string-equals-method-with-example/

   Observations:
   - "AllOperations" is the private member of the object. It applies the 3 asked methods within the object.
   - The above function has the filter method that returns numbers that are less than 25, a for loop that multiplies each element by 3 and stores its value in a new variable. Finally, we do a list comparison using the equals() method.



```scala
// Scala Program A
// filter method: https://www.geeksforgeeks.org/scala-list-filter-method-with-example/
// yield method: https://alvinalexander.com/scala/scala-for-loop-yield-examples-yield-tutorial/
// equals method: https://www.geeksforgeeks.org/scala-string-equals-method-with-example/
object HW4Question1ProgramA {
  private val list = List(5, 10, 15, 20, 25, 30) // chosen list
  private def AllOperations(): Unit = {
    val filteredList = list.filter(_ < 25) // return numbers that are less than 25
    println(s"Filtered list (numbers < 25): $filteredList")

    val multipliedList = for (e <- filteredList) yield e * 3 // for loop that multiplies each element by 3
    println(s"New list containing numbers from filtered list multiplied by 3: $multipliedList")

    val equalList = multipliedList.equals(List(15, 30, 45, 60)) // list comparison using equals() method
    println(s"EqualList == NewList? $equalList")
  }

  def main(args: Array[String]): Unit = {
    println(s"Chosen list for program A: $list")
    AllOperations()
  }
}
```



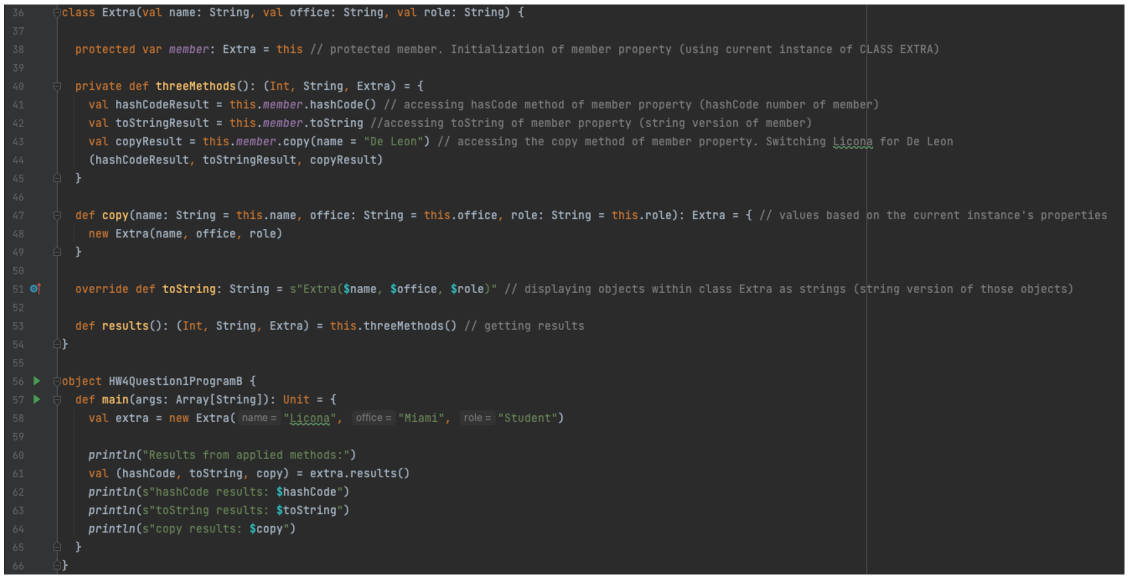
```
Chosen list for program A: List(5, 10, 15, 20, 25, 30)
Filtered list (numbers < 25): List(5, 10, 15, 20)
New list containing numbers from filtered list multiplied by 3: List(15, 30, 45, 60)
EqualList == NewList? true

Process finished with exit code 0
```

Program B
Resources:

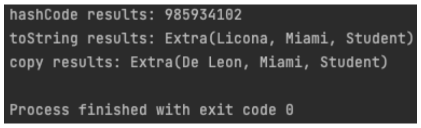
- Protected scope: https://www.geeksforgeeks.org/controlling-method-scope-in-scala/
- hashCode: https://www.geeksforgeeks.org/scala-string-hashcode-method-with-example/
- toString: https://www.geeksforgeeks.org/scala-int-tostring-method-with-example/
- class/copy: https://www.educative.io/answers/what-is-copy-in-scala
- this keyword: http://testingpool.com/scala-this-keyword/
- this keyword: https://www.geeksforgeeks.org/scala-this-keyword/
- overriding: https://stackoverflow.com/questions/16110370/scala-what-is-the-purpose-of-override

Observations:

- The following example came from the class/copy source. It's basically following the class Extra(name, office, and role) with the purpose of applying the 3 asked methods (specially the copy method).
- The private member can be access through the class Extra.
- The threeMethods function contains the three methods asked (hashCode, toString, and copy) and we are using the "this" keyword to access each method of the member property of the current instance. We are going to use the copy method to switch from Licona last name to De Leon last name.
- We have a copy function that is based on the default values as well as values based on the current instance's properties.
- We use the override method in the toString function to get the string version of the objects within the class Extra instead of the default values.
- Create result function to store results and run the object to get them.



```scala
class Extra(val name: String, val office: String, val role: String) {

  protected var member: Extra = this // protected member. Initialization of member property (using current instance of CLASS EXTRA)

  private def threeMethods(): (Int, String, Extra) = {
    val hashCodeResult = this.member.hashCode() // accessing hasCode method of member property (hashCode number of member)
    val toStringResult = this.member.toString //accessing toString of member property (string version of member)
    val copyResult = this.member.copy(name = "De Leon") // accessing the copy method of member property. Switching Licona for De Leon
    (hashCodeResult, toStringResult, copyResult)
  }

  def copy(name: String = this.name, office: String = this.office, role: String = this.role): Extra = { // values based on the current instance's properties
    new Extra(name, office, role)
  }

  override def toString: String = s"Extra($name, $office, $role)" // displaying objects within class Extra as strings (string version of those objects)

  def results(): (Int, String, Extra) = this.threeMethods() // getting results
}

object HW4Question1ProgramB {
  def main(args: Array[String]): Unit = {
    val extra = new Extra( name = "Licona",  office = "Miami",  role = "Student")

    println("Results from applied methods:")
    val (hashCode, toString, copy) = extra.results()
    println(s"hashCode results: $hashCode")
    println(s"toString results: $toString")
    println(s"copy results: $copy")
  }
}
```



```
hashCode results: 985934102
toString results: Extra(Licona, Miami, Student)
copy results: Extra(De Leon, Miami, Student)

Process finished with exit code 0
```

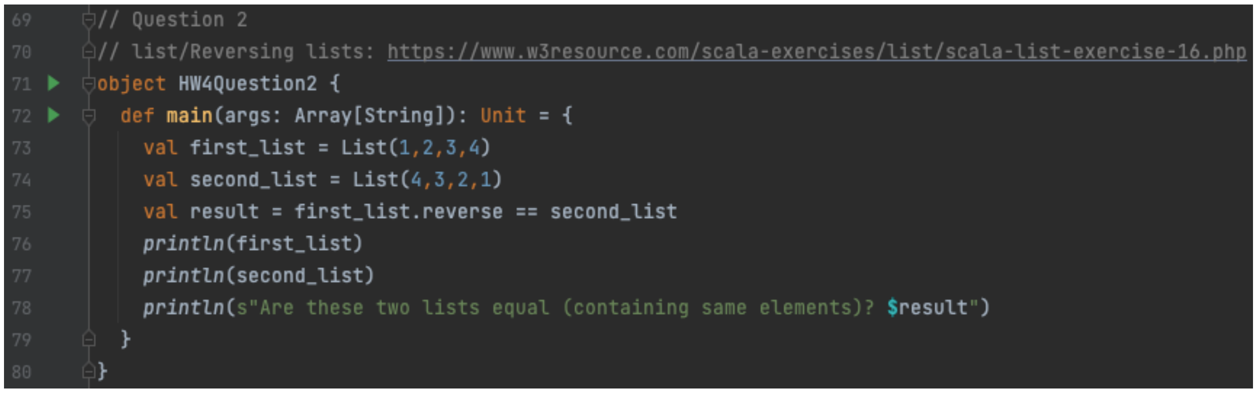
2. Question 2

   Resources (Please check Scala Worksheet Resources for more detail):

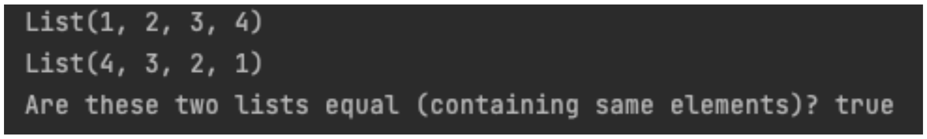
   - List/reverse: https://www.w3resource.com/scala-exercises/list/scala-list-exercise-16.php

   Observations:

   - This code is very self-explanatory, it has 2 lists and one variable call result which stores the results from the reverse comparison.



```scala
// Question 2
// list/Reversing lists: https://www.w3resource.com/scala-exercises/list/scala-list-exercise-16.php
object HW4Question2 {
  def main(args: Array[String]): Unit = {
    val first_list = List(1,2,3,4)
    val second_list = List(4,3,2,1)
    val result = first_list.reverse == second_list
    println(first_list)
    println(second_list)
    println(s"Are these two lists equal (containing same elements)? $result")
  }
}
```



```
List(1, 2, 3, 4)
List(4, 3, 2, 1)
Are these two lists equal (containing same elements)? true
```

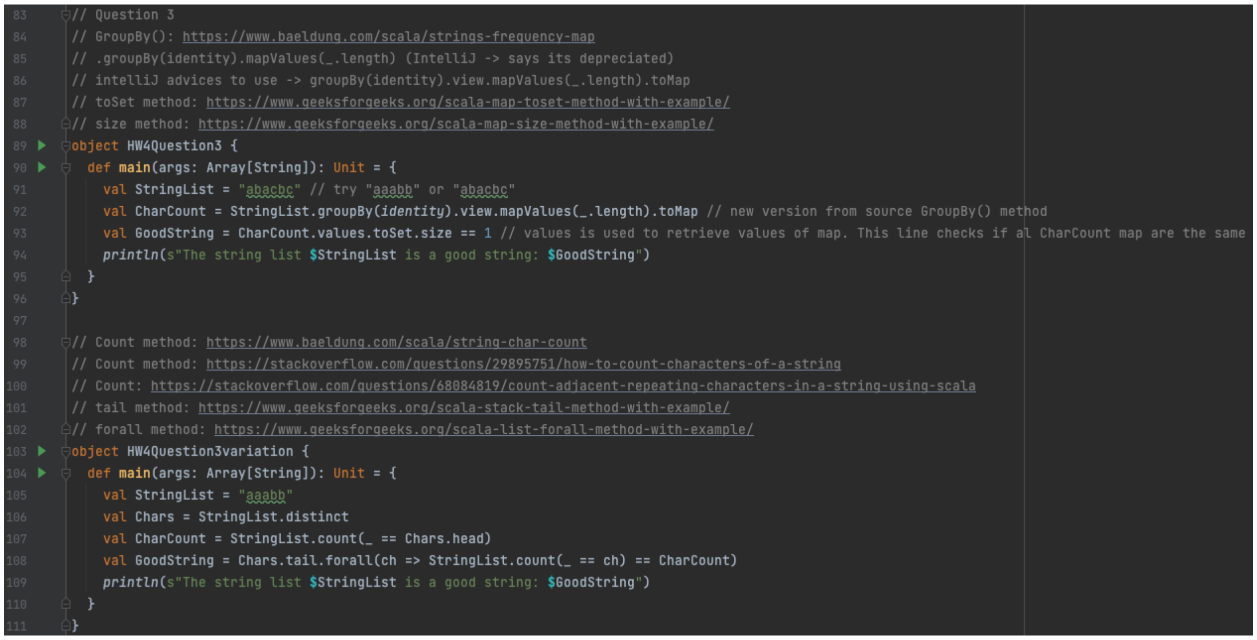
3. Question 3

   Resources:

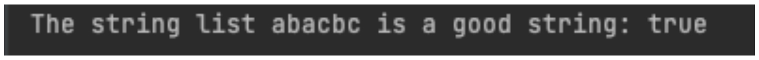
   - GroupBy: https://www.baeldung.com/scala/strings-frequency-map
   - toSet method: https://www.geeksforgeeks.org/scala-map-toset-method-with-example/
   - size method: https://www.geeksforgeeks.org/scala-map-size-method-with-example/

   Observations:

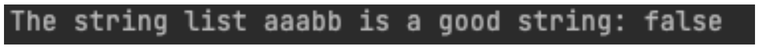
   - Just like question 2, this code is also very self-explanatory.
   - "After performing this step, we can group each character. The returned *Map* contains entries where for each entry, the key is each distinct character in the string, and the value is a list of all the occurrences of that character. Finally, we count how many occurrences happen in the text." (Group By source, check code comments).
   - I would also like to share another variation a tried for the same question names "HW4Question3variation".



```scala
// Question 3
// GroupBy(): https://www.baeldung.com/scala/strings-frequency-map
// .groupBy(identity).mapValues(_.length) (IntelliJ -> says its depreciated)
// intelliJ advices to use -> groupBy(identity).view.mapValues(_.length).toMap
// toSet method: https://www.geeksforgeeks.org/scala-map-toset-method-with-example/
// size method: https://www.geeksforgeeks.org/scala-map-size-method-with-example/
object HW4Question3 {
  def main(args: Array[String]): Unit = {
    val StringList = "abacbc" // try "aaabb" or "abacbc"
    val CharCount = StringList.groupBy(identity).view.mapValues(_.length).toMap // new version from source GroupBy() method
    val GoodString = CharCount.values.toSet.size == 1 // values is used to retrieve values of map. This line checks if al CharCount map are the same
    println(s"The string list $StringList is a good string: $GoodString")
  }
}

// Count method: https://www.baeldung.com/scala/string-char-count
// Count method: https://stackoverflow.com/questions/29895751/how-to-count-characters-of-a-string
// Count: https://stackoverflow.com/questions/68084819/count-adjacent-repeating-characters-in-a-string-using-scala
// tail method: https://www.geeksforgeeks.org/scala-stack-tail-method-with-example/
// forall method: https://www.geeksforgeeks.org/scala-list-forall-method-with-example/
object HW4Question3variation {
  def main(args: Array[String]): Unit = {
    val StringList = "aaabb"
    val Chars = StringList.distinct
    val CharCount = StringList.count(_ == Chars.head)
    val GoodString = Chars.tail.forall(ch => StringList.count(_ == ch) == CharCount)
    println(s"The string list $StringList is a good string: $GoodString")
  }
}
```



```
The string list abacbc is a good string: true
```



```
The string list aaabb is a good string: false
```

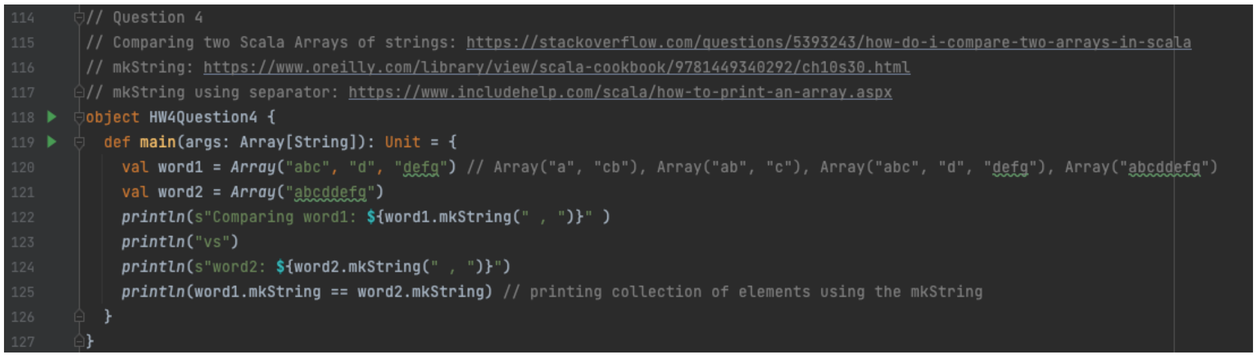
4. Question 4

   Resources:

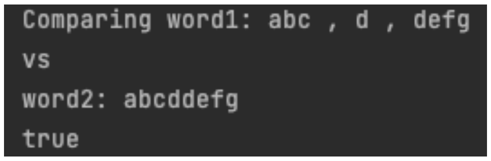
   - Comparing arrays of strings: https://stackoverflow.com/questions/5393243/how-do-i-compare-two-arrays-in-scala
   - Mkstring: https://www.oreilly.com/library/view/scala-cookbook/9781449340292/ch10s30.html
   - Mkstrings: https://www.includehelp.com/scala/how-to-print-an-array.aspx

   Observations:

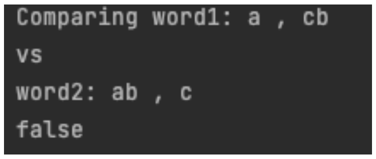
   - This last code is also simple and self-explanatory. We create 2 arrays, and we compare them using the mkString method and the == operator.



```scala
// Question 4
// Comparing two Scala Arrays of strings: https://stackoverflow.com/questions/5393243/how-do-i-compare-two-arrays-in-scala
// mkString: https://www.oreilly.com/library/view/scala-cookbook/9781449340292/ch10s30.html
// mkString using separator: https://www.includehelp.com/scala/how-to-print-an-array.aspx
object HW4Question4 {
  def main(args: Array[String]): Unit = {
    val word1 = Array("abc", "d", "defg") // Array("a", "cb"), Array("ab", "c"), Array("abc", "d", "defg"), Array("abcddefg")
    val word2 = Array("abcddefg")
    println(s"Comparing word1: ${word1.mkString(" , ")}" )
    println("vs")
    println(s"word2: ${word2.mkString(" , ")}")
    println(word1.mkString == word2.mkString) // printing collection of elements using the mkString
  }
}
```



```
Comparing word1: abc , d , defg
vs
word2: abcddefg
true
```



```
Comparing word1: a , cb
vs
word2: ab , c
false
```

```scala
// Question 1

// Scala Program A

object HW4Question1ProgramA {

  private val list = List(5, 10, 15, 20, 25, 30) // chosen list

  private def AllOperations(): Unit = {

    val filteredList = list.filter(_ < 25) // return numbers that are less than 25

    println(s"Filtered list (numbers < 25): $filteredList")


    val multipliedList = for (e <- filteredList) yield e * 3 // for loop that multiplies each element by 3

    println(s"New list containing numbers from filtered list multiplied by 3: $multipliedList")


    val equalList = multipliedList.equals(List(15, 30, 45, 60)) // list comparison using equals() method

    println(s"EqualList == NewList? $equalList")

  }


  def main(args: Array[String]): Unit = {

    println(s"Chosen list for program A: $list")

    AllOperations()

  }

}
```

```
// Scala Program B

class Extra(val name: String, val office: String, val role: String) {

  protected var member: Extra = this // protected member. Initialization of member property (using
current instance of CLASS EXTRA)

  private def threeMethods(): (Int, String, Extra) = {

    val hashCodeResult = this.member.hashCode() // accessing hasCode method of member property
(hashCode number of member)

    val toStringResult = this.member.toString //accessing toString of member property (string version of
member)

    val copyResult = this.member.copy(name = "De Leon") // accessing the copy method of member
property. Switching Licona for De Leon

    (hashCodeResult, toStringResult, copyResult)

  }

  def copy(name: String = this.name, office: String = this.office, role: String = this.role): Extra = { // values
based on the current instance's properties

    new Extra(name, office, role)

  }

  override def toString: String = s"Extra($name, $office, $role)" // displaying objects within class Extra as
strings (string version of those objects)

  def results(): (Int, String, Extra) = this.threeMethods() // getting results

}

object HW4Question1ProgramB {

  def main(args: Array[String]): Unit = {

    val extra = new Extra("Licona", "Miami", "Student")

    println("Results from applied methods:")

    val (hashCode, toString, copy) = extra.results()

    println(s"hashCode results: $hashCode")

    println(s"toString results: $toString")

    println(s"copy results: $copy")

  }

}
```

```scala
// Question 2

object HW4Question2 {

  def main(args: Array[String]): Unit = {

    val first_list = List(1,2,3,4)

    val second_list = List(4,3,2,1)

    val result = first_list.reverse == second_list

    println(first_list)

    println(second_list)

    println(s"Are these two lists equal (containing same elements)? $result")

  }

}
```

```
// Question 3

object HW4Question3 {

  def main(args: Array[String]): Unit = {

    val StringList = "abacbc" // try "aaabb" or "abacbc"

    val CharCount = StringList.groupBy(identity).view.mapValues(_.length).toMap // new version from
source GroupBy() method

    val GoodString = CharCount.values.toSet.size == 1 // values is used to retrieve values of map. This line
checks if al CharCount map are the same

    println(s"The string list $StringList is a good string: $GoodString")

  }

}



object HW4Question3variation {

  def main(args: Array[String]): Unit = {

    val StringList = "aaabb"

    val Chars = StringList.distinct

    val CharCount = StringList.count(_ == Chars.head)

    val GoodString = Chars.tail.forall(ch => StringList.count(_ == ch) == CharCount)

    println(s"The string list $StringList is a good string: $GoodString")

  }

}
```

```
// Question 4

object HW4Question4 {

  def main(args: Array[String]): Unit = {

    val word1 = Array("abc", "d", "defg") // Array("a", "cb"), Array("ab", "c"), Array("abc", "d", "defg"),
Array("abcddefg")

    val word2 = Array("abcddefg")

    println(s"Comparing word1: ${word1.mkString(" , ")}" )

    println("vs")

    println(s"word2: ${word2.mkString(" , ")}")

    println(word1.mkString == word2.mkString) // printing collection of elements using the mkString

  }

}
```