

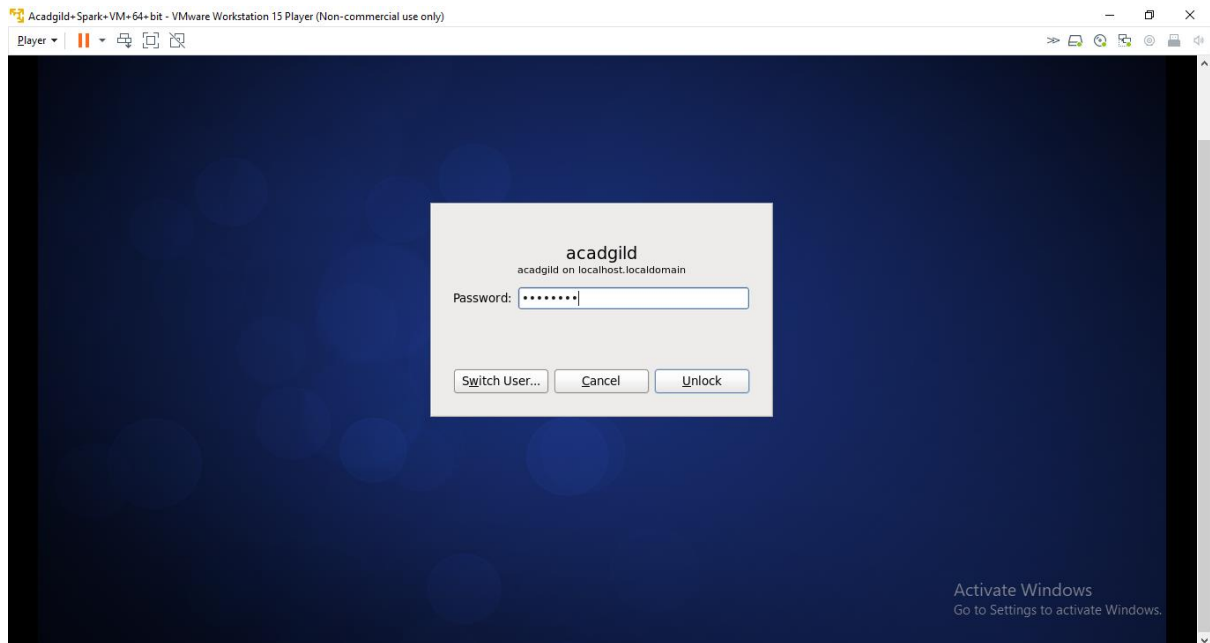
Acadgild-Assignement-43:

Problem Statement:

Task 1

Follow the below link document steps to download and import Acadgild Spark VM in the Oracle Virtual Box.

Answer: - Done and attached screen-shot the VM.



Task 2

Given a list of strings - List[String] ("alpha", "gamma", "omega", "zeta", "beta")

Q 2.1: find count of all strings with length 4.

Answer: - Here I have used the lambda function that evaluate the given condition, then list.count method count the total number of cases that are true.

```
Acadgild+Spark+VM+64-bit - VMware Workstation 15 Player (Non-commercial use only)
Player
Applications Places System
acadmild@localhost:~
File Edit View Search Terminal Help
Type in expressions to have them evaluated.
Type :help for more information.

scala> scala> // Given a List of String ["Alpha","Gama","omega","beta","zeta"]
// Detected repl transcript paste: ctrl-D to finish.
// Replaying 1 commands from transcript.
scala> // Given a List of String ["Alpha","Gama","omega","beta","zeta"]

scala> //Defined The List
scala> var list= List("Alpha","Gama","omega","beta","zeta")
list: List[String] = List(Alpha, Gama, omega, beta, zeta)

scala> //Now defining the lambda expression to claculate the count of string wit
h length =4
scala> var count:Int=list.count(s=>s.length==4)
count: Int = 3

scala> █
```

Q2.2:- Convert the list of string to a list of integers, where each string is mapped to its Corresponding length.

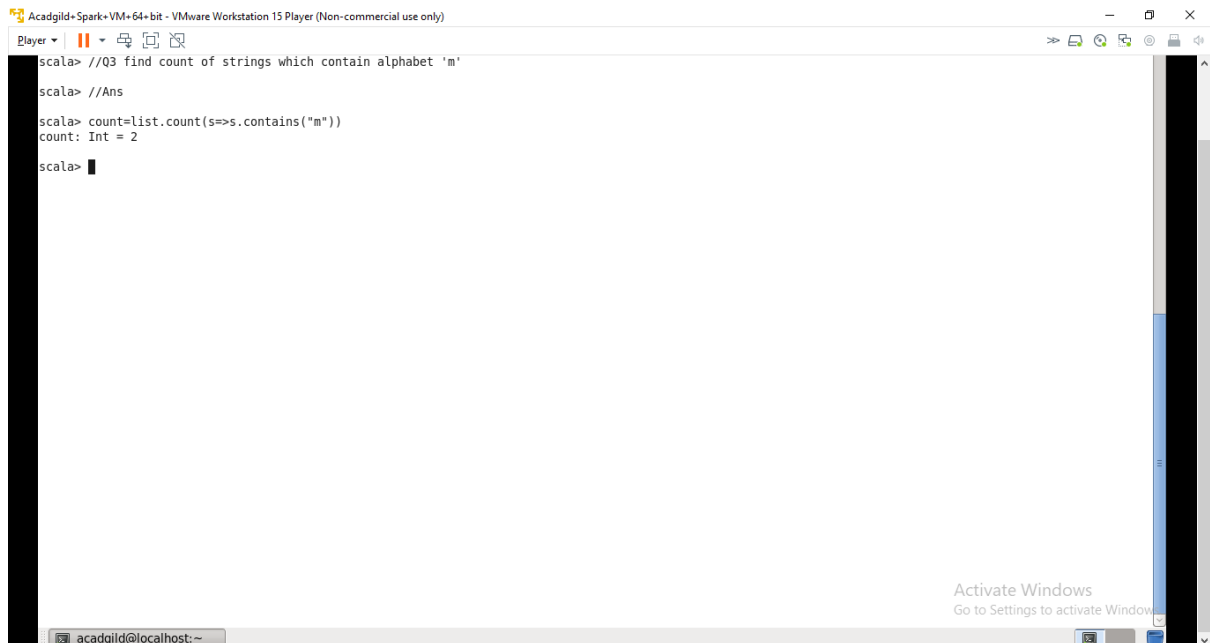
Answer: - Here I have used the lambda function that count the length of each string in list which is then mapped by its length using the list.map method.

```
scala> //Q2
scala> //Convert the list of String into list of integer ,where each string is mapped to its corresponding length
scala> var intlist=list.map(s=>s.length)
intlist: List[Int] = List(5, 4, 5, 4, 4)

scala> █
```

Q2.3: Find the count of strings that contains the alphabet 'm'.

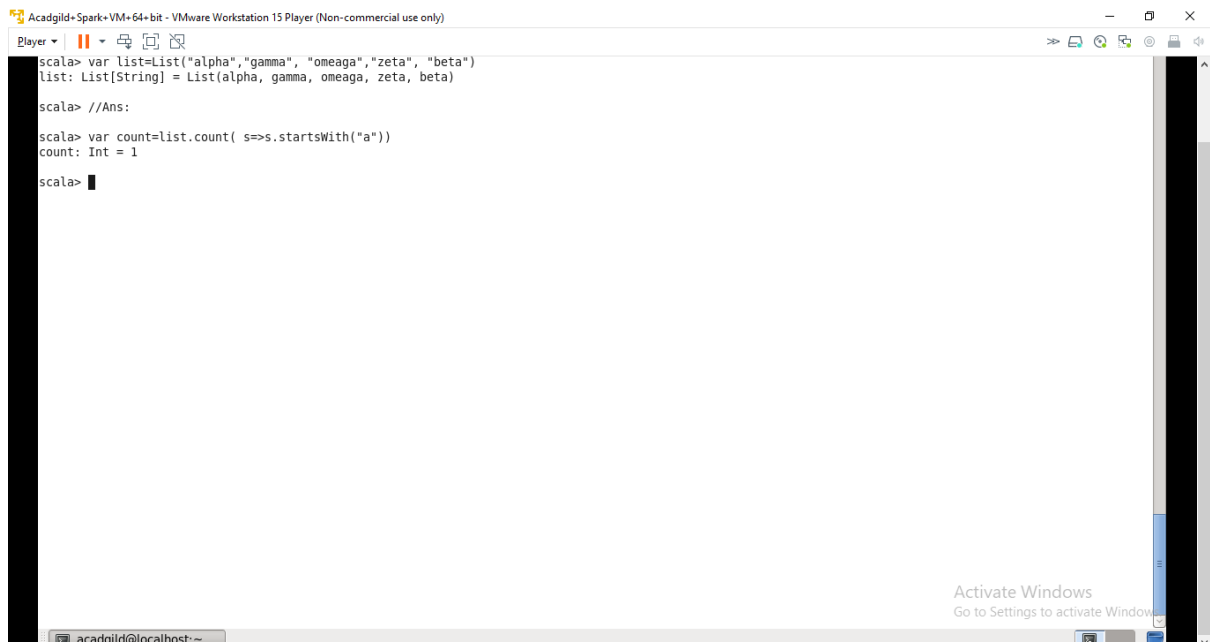
Answer: - Here I have used the lambda function that evaluate the given condition, then list.count method count the total number of cases that are true.



```
Acadgild+Spark+VM+64+bit - VMware Workstation 15 Player (Non-commercial use only)
Player
scala> //Q3 find count of strings which contain alphabet 'm'
scala> //Ans
scala> count=list.count(s=>s.contains("m"))
count: Int = 2
scala>
```

Q2.4:- Find the count of all strings which start with the alphabet 'a'.

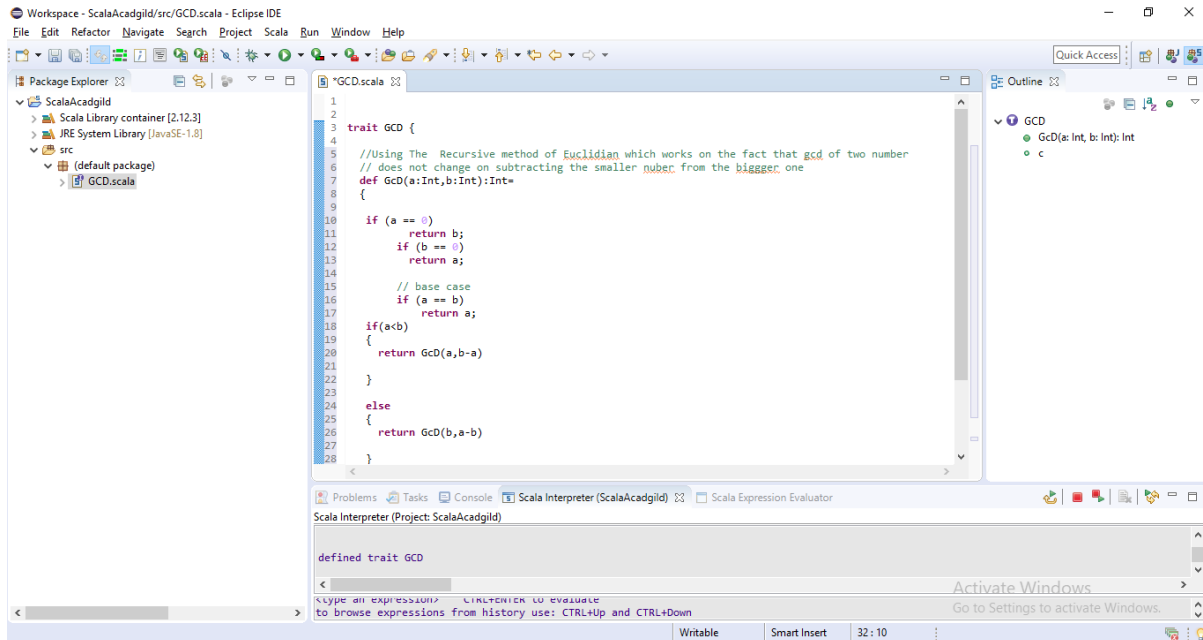
Answer: - Here I have used the lambda function that evaluate the given condition, then list.count method count the total number of cases that are true.



```
Acadgild+Spark+VM+64+bit - VMware Workstation 15 Player (Non-commercial use only)
Player
scala> var list=List("alpha","gamma", "omeaga","zeta", "beta")
list: List[String] = List(alpha, gamma, omeaga, zeta, beta)
scala> //Ans:
scala> var count=list.count( s=>s.startsWith("a"))
count: Int = 1
scala>
```

Q3:- Create a Scala application to find the GCD of two numbers.

Answer:- An **efficient solution** is to use Euclidean algorithm which is the main algorithm used for this purpose. The idea is, GCD of two numbers doesn't change if smaller number is subtracted from a bigger number.



The screenshot displays the Eclipse IDE interface with a Scala project named 'ScalaAcadgild'. The main editor shows the file 'GCD.scala' containing the following code:

```
1
2
3 trait GCD {
4
5 //Using The Recursive method of Euclidian which works on the fact that gcd of two number
6 // does not change on subtracting the smaller number from the bigger one
7 def Gcd(a:Int,b:Int):Int=
8 {
9
10 if (a == 0)
11     return b;
12 if (b == 0)
13     return a;
14
15 // base case
16 if (a == b)
17     return a;
18 if(a<b)
19 {
20     return Gcd(a,b-a)
21 }
22
23 else
24 {
25     return Gcd(b,a-b)
26 }
27 }
28 }
```

The Package Explorer on the left shows the project structure with 'GCD.scala' under the 'src' folder. The Outline view on the right shows the 'GCD' trait and its 'Gcd' method. The Scala Interpreter console at the bottom shows the output: 'defined trait GCD'. The status bar at the bottom indicates 'Writable', 'Smart Insert', and '32:10'.