ΟΔΗΓΙΕΣ ΓΙΑ ΤΗΝ ΔΗΜΙΟΥΡΓΙΑ ΕΙΚΟΝΙΚΗΣ ΜΗΧΑΝΗΣ LINUX (UBUNTU) ΚΑΙ ΕΓΚΑΤΑΣΤΑΣΗ ΤΟΥ ΛΟΓΙΣΜΙΚΟΥ SPARK

Σε όσα ακολουθούν παρουσιάζονται οδηγίες για την δημιουργία μιας εικονικής μηχανής Linux και συγκεκριμένα Ubuntu και την εγκατάσταση του λογισμικού SPARK.

Για την δημιουργία και την διαχείριση της εικονικής μηχανής έχει χρησιμοποιηθεί το λογισμικό Oracle VM VirtualBox και συγκεκριμένα η έκδοση 7.0. Η εγκατάσταση έγινε σε ένα σταθμό εργασίας με λειτουργικό σύστημα Windows10 64bit.

Παρατήρηση: Υπάρχουν διάφοροι τρόποι για να εγκαταστήσετε και να παραμετροποιήσετε ένα περιβάλλον στο οποίο μπορείτε να τρέξετε το λογισμικό Spark. Οι παρακάτω οδηγίες περιγράφουν τα βήματα που ακολουθήθηκαν για την δημιουργία της εικονικής μηχανής που χρησιμοποιήθηκε στο σεμινάριο του μαθήματος. Μπορείτε να τα ακολουθήσετε για να δημιουργήσετε την δική σας εικονική μηχανή.

1. Εγκατάσταση του λογισμικού Virtual Box

Από τον παρακάτω σύνδεσμο κατεβάζουμε και εγκαθιστούμε το λογισμικό VirtualBox. Η τελευταία έκδοση είναι η **7.0**

https://www.virtualbox.org/

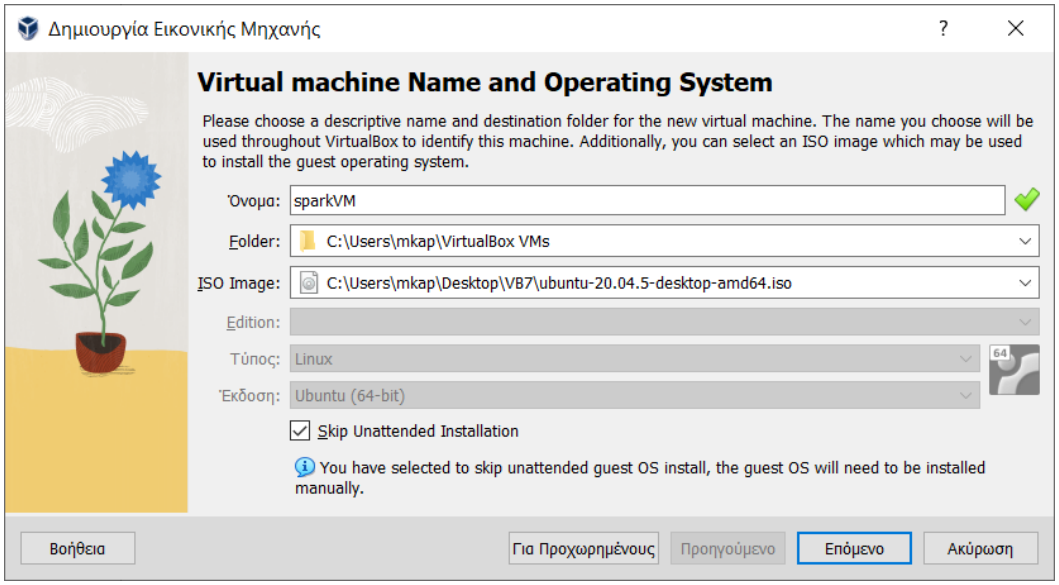
2. Download Ubuntu-20.04.3scala:/o

Από τον παρακάτω σύνδεσμο κατεβάζουμε το λειτουργικό Ubuntu:

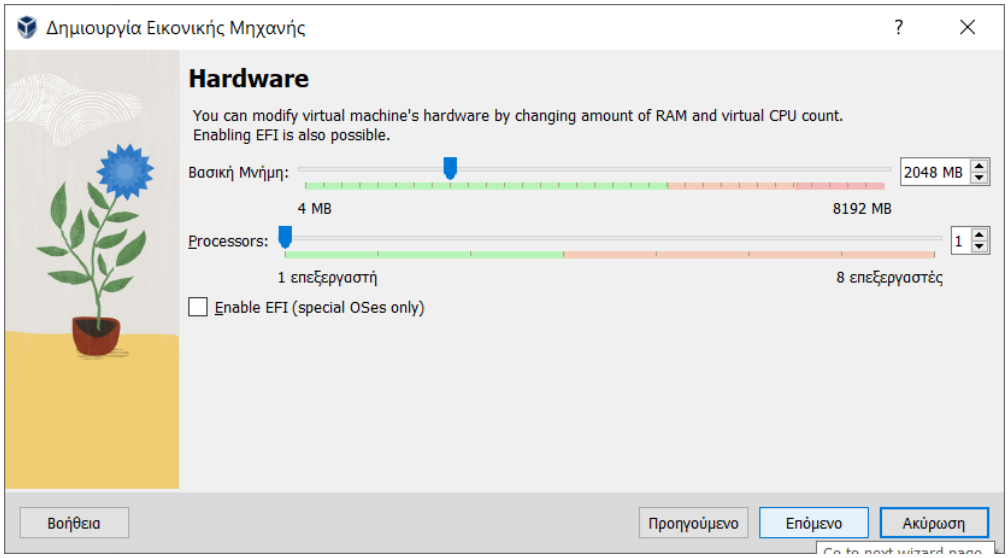
https://releases.ubuntu.com/jammy/

Επιλέξτε το image**: ubuntu-22.04.3-desktop-amd64.iso**

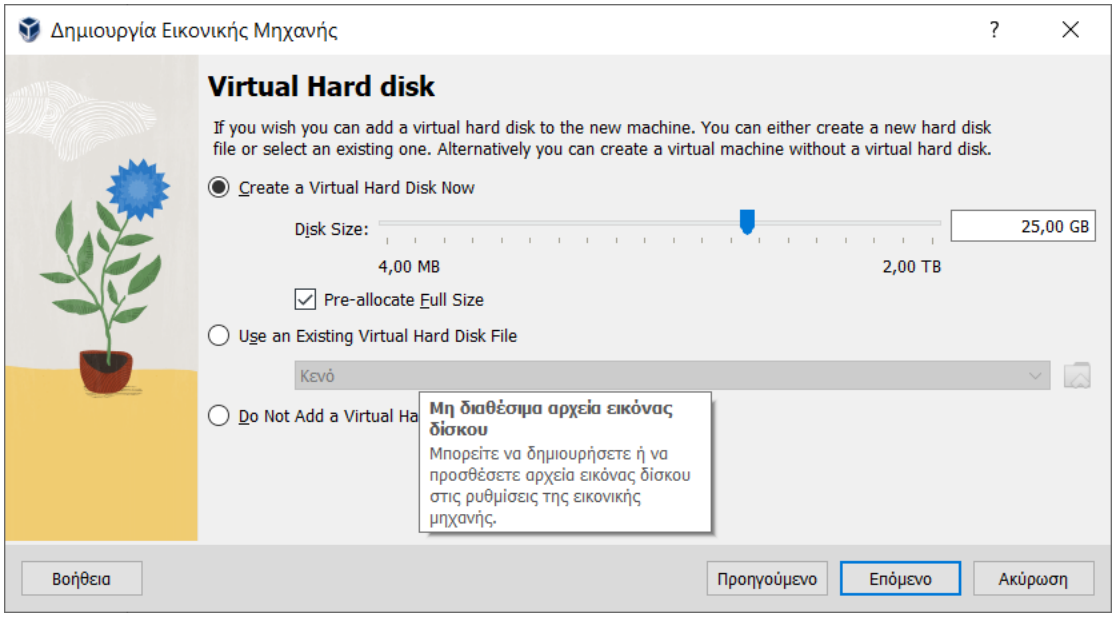
3. Ξεκινάμε το VirtualBox και από το μενού επιλέγουμε Μηχανή-->Νέα στο παράθυρο που εμφανίζεται δίνουμε τα στοιχεία της εικονικής μηχανής που θα δημιουργήσουμε. Συγκεκριμένα προσδιορίζουμε το Ονομα, τον φάκελο στον οποίο θα δημιουργηθούν τα αρχεία της εικονικής μηχανής και επιλέγουμε το IMAGE του Ubuntu που έχουμε κατεβάσει. ΠΡΟΣΟΧΗ: επιλέξτε “Skip Unattended Installation”.



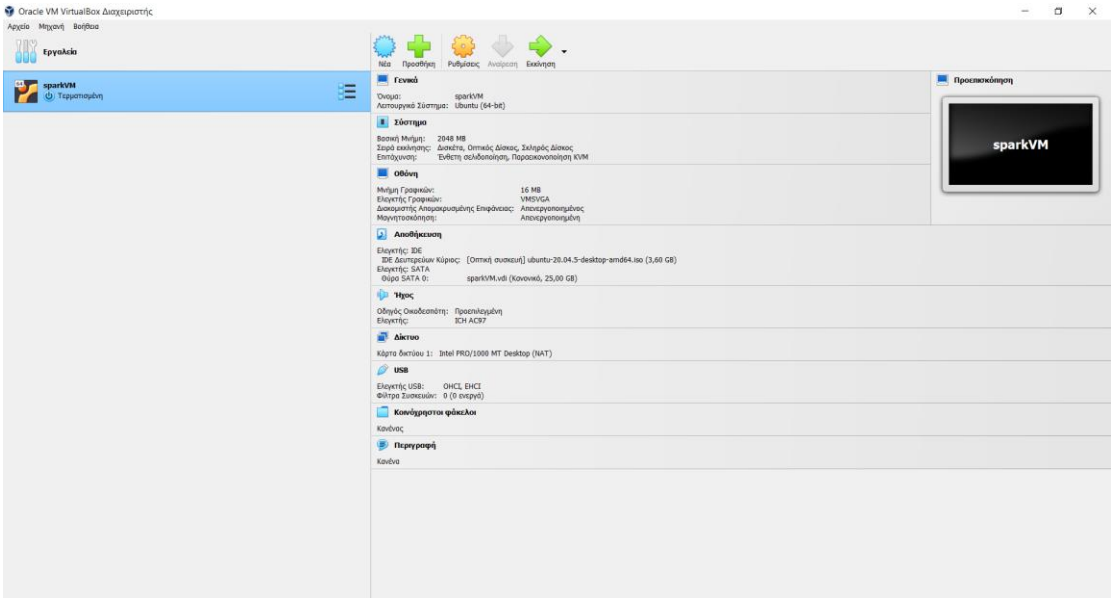
Στο επόμενο βήμα ορίζουμε το μέγεθος της μνήμης (προτεινόμενο 2048MB)



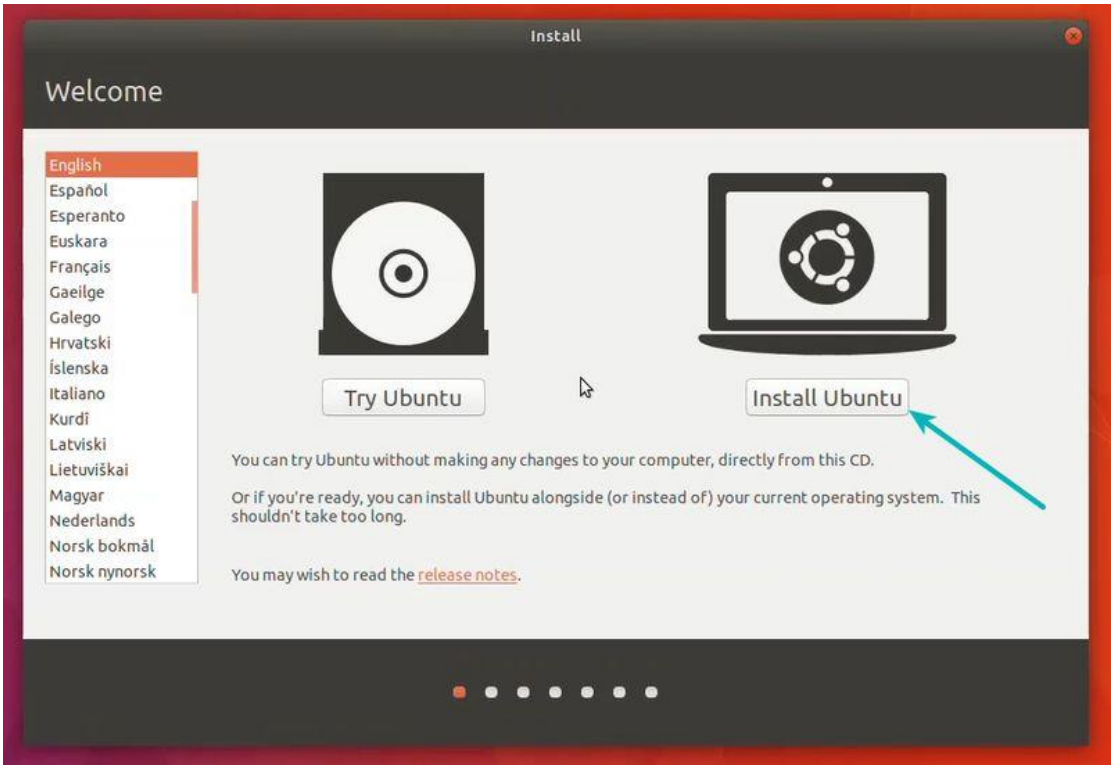
Στην οθόνη με τις ρυθμίσεις του χώρου επιλέξτε «Pre-allocate Full Size».



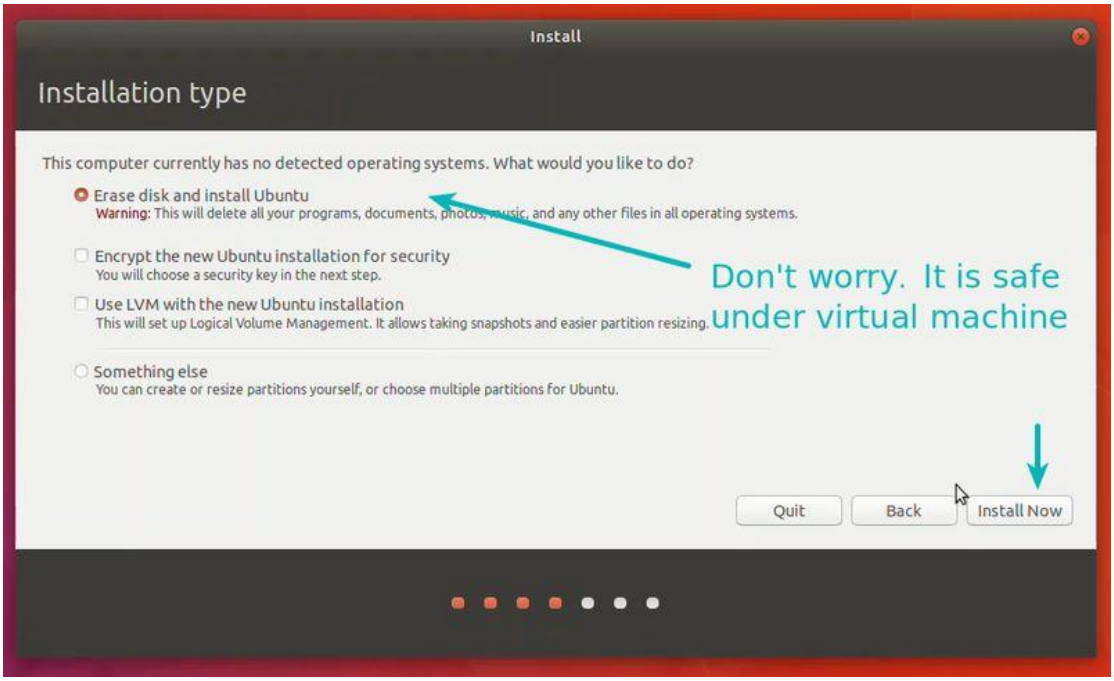
Στο σημείο αυτό επιλέγετε την εικονική μηχανή που δημιουργήσατε και κάνετε κλικάρετε στο πράσινο βελάκι (εκκίνηση) ώστε να ξεκινήσει η εικονική μηχανή.

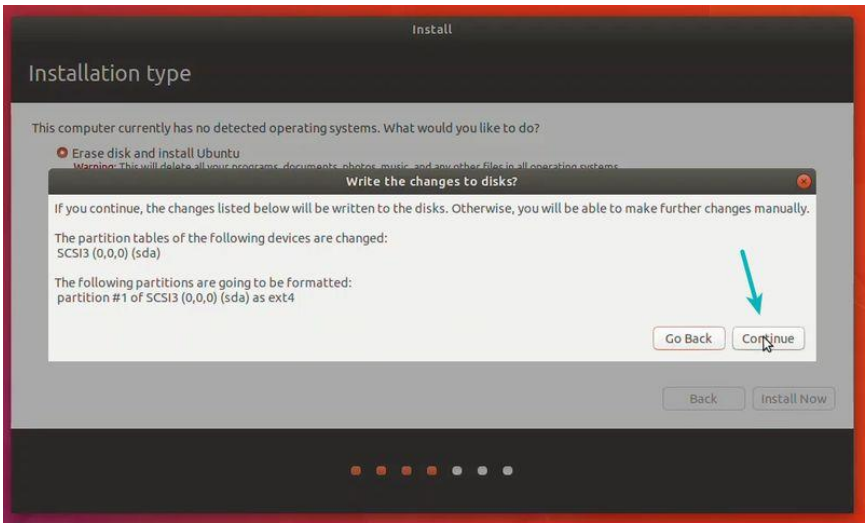


Η εγκατάσταση του λειτουργικού Ubuntu-20.04 θα ξεκινήσει. Μόλις εμφανιστεί η παρακάτω οθόνη επιλέξτε **install Ubuntu.**

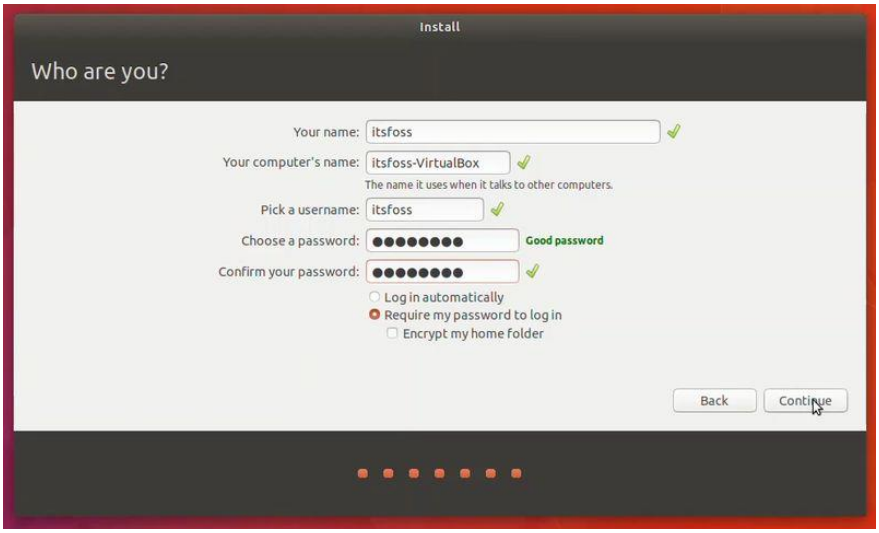


Μην ανησυχείτε δεν πρόκειται να διαγραφούν τα αρχεία σας από το PC. Η διαμόρφωση αφορά τον "δίσκο" της εικονικής μηχανής.





Στο σημείο αυτό πρέπει να ορίσετε τα στοιχεία σύνδεσης (login) στο Linux.



Τα χαρακτηριστική της εικονικής μηχανής που χρησιμοποιήθηκε στο εργαστήριο είναι:

your name = lab

computer name = pclab

username = lab

password = lab2023 (είναι σημαντικό να ορίσετε ένα συνθηματικό)

Ακολουθεί η εγκατάσταση του συστήματος και στο τέλος γίνεται επανεκκίνηση.

Η εικονική μηχανή είναι έτοιμη. Στο διάλογο του login θα δείτε το όνομα του χρήστη (lab) θα πληκτρολογήσετε το συνθηματικό (lab2023) και θα αποκτήσετε πρόσβαση στο λειτουργικό Ubuntu. Αν ανοίξετε ένα τερματικό (terminal) και πληκτρολογήσετε την εντολή pwd θα εμφανιστεί το path του home directory (/home/lab) στην συγκεκριμένη περίπτωση.

4. Εισαγωγή Δίσκου με προσθήκες επισκέπτη

Κατόπιν της εγκατάστασης του ubuntu από το μενού επιλογών της εικονικής μηχανής επιλέγουμε **Συσκευές --> Εισαγωγή Δίσκου με τις προσθήκες επισκέπτη**. Στην συνέχεια κάνουμε επανεκκίνηση της εικονικής μηχανής. Αυτό για να έχουμε καλύτερη προσαρμογή της επιφάνειας εργασίας καθώς επίσης και δυνατότητες copy-paste μεταξύ της εικονικής μηχανής και του συστήματός μας (π.χ. windows). Απαιτείται επίσης να ενεργοποιήσουμε την επιλογή **Συσκευές -- Κοινόχρηστα πρόχειρα --> Bidirectional**

6. Εγκατάσταση SPARK

6.1. Εγκατάσταση προαπαιτούμενων πακέτων

Πριν εγκαταστήσουμε το λογισμικό SPARK πρέπει να εγκαταστήσουμε ορισμένα απαιτούμενα πακέτα και συγκεκριμένα:

- **JDK**
- **Scala**
- **Git**

Ανοίγουμε ένα terminal και από περιβάλλον φλοιού (Shell) και το home directory (/home/lab στην περίπτωσή μας) δίνουμε την εντολή:

```
$ sudo apt update [ενημέρωση πακέτων Ubuntu]
$ sudo apt install default-jdk scala git -y
```

Μπορείτε να δείτε την έκδοση των πακέτων που εγκαταστάθηκαν με τις παρακάτω εντολές:

```
$ java -version    [11.0.20.1]
$ javac -version
$ scala -version   [2.11.12]
$ git --version
```

6.2. Εγκατάσταση του Spark 3.5.0

Δίνουμε την εντολή

```
$ wget https://dlcdn.apache.org/spark/spark-3.5.0/spark-3.5.0-bin-hadoop3.tgz
```

Στον φάκελο εργασίας (/home/lab) θα κατέβει το αρχείο **`spark-3.5.0-binhadoop3.tgz`**

Στη συνέχεια κάνουμε extract τα αρχεία με την εντολή:

```
$ tar xvf spark-*
```

Δημιουργείται ο φάκελος **`spark-3.5.0-binhadoop3`** τον οποίο θα μετακινήσουμε και θα μετονομάσου με εντολή **mv** ως εξής:

```
$ sudo mv spark-3.5.0-binhadoop3 /opt/spark
```

6.3. Διαμόρφωση Περιβάλλοντος

Πριν ξεκινήσουμε το Spark πρέπει να διαμορφώσουμε κατάλληλα το περιβάλλον ορίζοντας τις εξής μεταβλητές **στο τέλος** του αρχείου **.profile** το οποίο βρίσκεται στο home directory του χρήστη.

```
export SPARK_HOME=/opt/spark
export PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin:.
export PYSPARK_PYTHON=/usr/bin/python3
```

Μπορούμε να προσθέσουμε τις παραπάνω μεταβλητές χρησιμοποιώντας έναν editor (π.χ. vi, nano, text editor).

Όταν ολοκληρώσουμε τις αλλαγές εκτελούμε το αρχείο **.profile** ως εξής:

```
$ source ~/.profile (η αποσυνδεόμαστε και ξανακάνουμε login)
```

6.4. Εκκίνηση του Spark

Εκκίνηση του Master (Standalone Spark Master Server)

```
$ start-master.sh
```

Μπορούμε να δούμε την web διεπαφή του Server με έναν από τους παρακάτω τρόπους:

```
$ http://127.0.0.1:8080/
$ localhost:8080
$ pclab:8080    (devicename:8080)
```

Εκκίνηση του Slave (Start a Worker Process)

```
start-worker.sh spark://master:port
```

Στην περίπτωσή μας:

```
start-worker.sh spark://pclab:7077
```
Εξ ορισμού η διεργασία worker χρησιμοποιεί όλους τους διαθέσιμους πυρήνες της cpu.

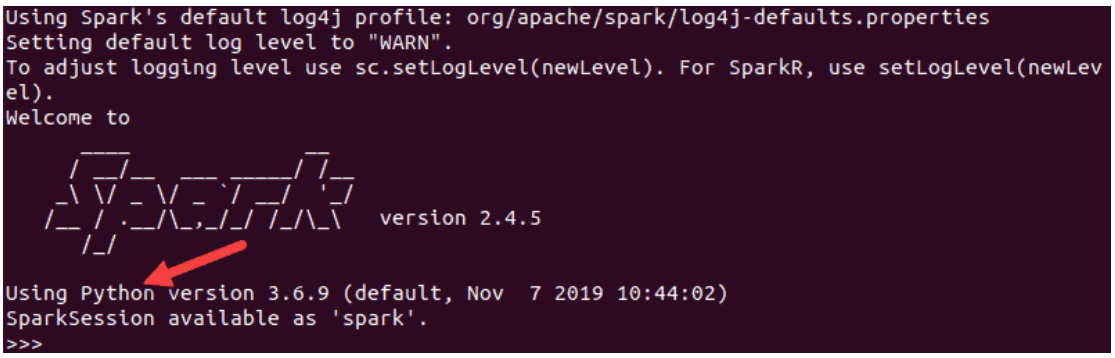
```
start-worker.sh -c 1 spark://pclab:7077
```
(θέλουμε η διεργασία worker να χρησιμοποίει έναν πυρήνα της CPU)

```
start-worker.sh -m 512M spark://pclab:7077
```
(Θέλουμε η διεργασία worker να χρησιμοποιεί 512MB RAM)

Δοκιμή του Spark Shell

```
$ spark-shell (SCALA)
```


```
$ pyspark (PYTHON)
```



Stop Master and Slave

```
$ stop-master.sh
$ stop-worker.sh
```

7. Εγκατάσταση του εργαλείου SBT (Scala Build Tool) version 1.9

Στο σημείο αυτό θα ανοίξετε ένα τερματικό (terminal) και από το φλοιό ($) θα δώσετε τις παρακάτω εντολές:

$ sudo apt install curl

```
$ echo "deb https://repo.scala-sbt.org/scalasbt/debian all main" |
sudo tee /etc/apt/sources.list.d/sbt.list

$ echo "deb https://repo.scala-sbt.org/scalasbt/debian /" | sudo tee
/etc/apt/sources.list.d/sbt_old.list

$ curl -sL
"https://keyserver.ubuntu.com/pks/lookup?op=get&search=0x2EE0EA64E40A
89B84B2DF73499E82A75642AC823" | sudo apt-key add
```

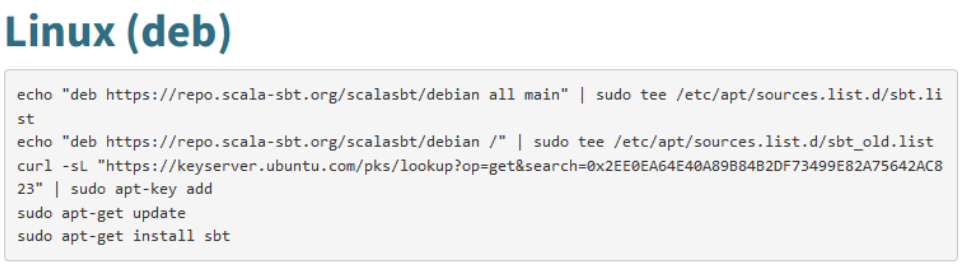
(Προσοχή: είναι μια συνεχόμενη εντολή)

```
$ sudo apt-get update

$ sudo apt-get install sbt
```

ΠΡΟΣΟΧΗ: Μπορείτε να βρείτε αυτές τις εντολές στον παρακάτω σύνδεσμο:

https://www.scala-sbt.org/download.html



Για να δείτε αν η εγκατάσταση έγινε με επιτυχία πληκτρολογείστε

$ sbt

Την πρώτη φορά που τρέχει το sbt θα εγκαταστήσει ορισμένες βιβλιοθήκες και στην συνέχεια θα σας βγάλει στο παρακάτω prompt από όπου μπορείτε να βγείτε με exit.

sbt:lab>exit
Στο home directory έχουν δημιουργηθεί οι φάκελοι project και target.

8. Εγκατάσταση των βιβλιοθηκών matplotlib και pandas

Μπορείτε να εγκαταστήσετε τις βιβλιοθήκες matplotlib και pandas που χρησιμοποιήσαμε στο εργαστήριο για να κάνουμε plot ως εξής:

```
$ sudo apt-get install python3-matplotlib
$ sudo apt-get install python3-pandas
```

9. Δημιουργία Spark Project με το εργαλείο SBT

Με το εργαλείο SBT έχουμε την δυνατότητα να κάνουμε να μεταγλωττίσουμε μια εφαρμογή και να δημιουργήσουμε το αρχείο .jar

Έστω μια απλή εφαρμογή η οποία έχει γραφεί σε scala και ο κώδικας βρίσκεται στο αρχείο wordCount.scala.

```
/* wordCount.scala */
import org.apache.spark._
import org.apache.spark.SparkContext._

object wordCount {
    def main(args: Array[String]) {
```

```
      val inputFile = args(0)
      val outputFile = args(1)
      val conf = new SparkConf().setAppName("wordCount")
      // Create a Scala Spark Context.
      val sc = new SparkContext(conf)
      // Load our input data.
      //val input =  sc.textFile(inputFile)
      val input =
sc.textFile("file:///home/lab/myprj/wordCount/"+inputFile)
      // Split up into words.
      val words = input.flatMap(line => line.split(" "))
      // Transform into word and count.
      val counts = words.map(word => (word, 1)).reduceByKey ( (x, y)
=> x + y)
      // Save the word count back out to a text file, causing
evaluation.

counts.saveAsTextFile("file:///home/lab/myprj/wordCount/"+outputFile)
    }
}
```

Πρέπει να δημιουργήσουμε ένα αρχείο build.sbt (configuration file) στο οποίο να ορίσουμε τις εξαρτήσεις (library dependencies) καθώς η εφαρμογή μας χρησιμοποιεί το API του SPARK. Το αρχείο build.sbt έχει την παρακάτω μορφή:

```
name := "wordCount"
version := "1.0"
scalaVersion := "2.12.11"

libraryDependencies += "org.apache.spark" % "spark-core_2.12" % "3.0.1"
```

Το εργαλείο sbt για να λειτουργήσει σωστά απαιτεί την δημιουργία μιας συγκεκριμένης ιεραρχικής δομής καταλόγων.

Έστω ότι ο κατάλογος εργασίας μας είναι :

/home/lab/myprj/wordCount

Μέσα στον κατάλογο wordCount πρέπει να υπάρχει το αρχείο build.sbt. Στην συνέχεια δημιουργούμε την εξής ιεραρχική δομή καταλόγων:

/home/lab/myprj/wordCount
/home/lab/myprj/wordCount/src
/home/lab/myprj/wordCount/src/main
/home/lab/myprj/wordCount/src/main/scala

Το αρχείο με τον κώδικα της εφαρμογής "wordCount.scala" πρέπει να βρίσκεται στον κατάλογο /home/lab/myprj/wordCount/src/main/scala

Πηγαίνουμε στον κατάλογο /home/lab/myprj/wordCount και δίνουμε την εντολή:

sbt package

Η εντολή αυτή θα δημιουργήσει τους καταλόγους project και target μέσα στον φάκελο wordCount και θα παράγει το αρχείο .jar στον κατάλογο wordCount/target/scala-2.12

Στην συνέχεια από τον φάκελο wordCount μπορούμε να τρέξουμε την εφαρμογή ως εξής:

```
/opt/spark/bin/spark-submit   --class "wordCount"   --master local[1]   target/scala-2.12/wordcount_2.12-1.0.jar
```