ASSIGNMENT E1

Spark Example for CS4B

Submitted By: Ahmed Dider Rahat

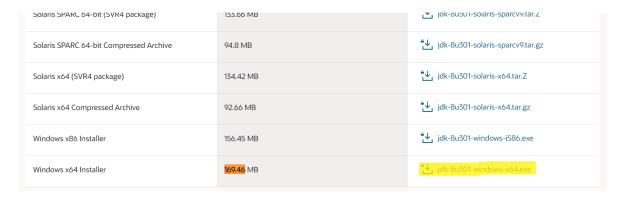
Matriculation Number: 916146

<u>Assignment Code:</u> After implementing all the assignment I pushed my code on my <u>Github link</u>.

Answer to the question no. 1

I have completed the installation process by using some steps. They are:

1. Java Installation: I had already installed java 17 in my system. So, at first I remove my version of java and then install java 8 from https://www.oracle.com/java/technologies/javase/javase8u211-later-archive-downloads.html. There are several version of jdk uploaded. I choose the marked one.



So, the current java version become:

```
C:\Users\DELL>java -version
java version "1.8.0_301"
Java(TM) SE Runtime Environment (build 1.8.0_301-b09)
Java HotSpot(TM) 64-Bit Server VM (build 25.301-b09, mixed mode)
```

2. Python Installation: Python 3 already installed before. So, just check the version:

```
C:\Users\DELL>python --version
Python 3.6.5 :: Anaconda, Inc.
```

- **3. Folder Creation:** Create two empty folder in C drive. One is Spark and another one is Hadoop/bin.
- **4. Download Spark:** Download spark from https://spark.apache.org/downloads.html and download the spark zip file.

Download Apache Spark™

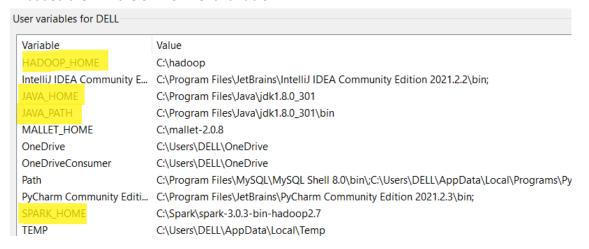
1. Choose a Spark release: 3.0.3 (Jun 23 2021) 🗸
2. Choose a package type: Pre-bu	uilt for Apache Hadoop 2.7
3 Download Spark: spark_3 0 3-bi	n-hadoon27tgz

4. Verify this release using the 3.0.3 signatures, checksums and project release KEYS.

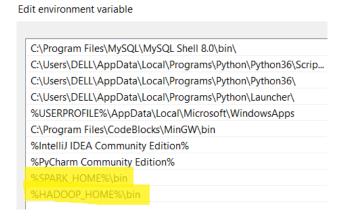
Note that Spark 3 is pre-built with Scala 2.12 in general and Spark 3.2+ provides additional pre-built distribution with Scala 2.13.

After download the zip file, I unzip it and move it to the C:\Spark location.

- **5. Download winutils:** I download the winutils file from github. As I used hadoop 2.7 my link was https://github.com/cdarlint/winutils/tree/master/hadoop-2.7.7/bin. Then I move it to C:\hadoop\bin.
- **6.** Setup environment variables: For the setting I need to setup the environment variables. So, I added them in the environment variable.



Then add the path of spark and hadoop in the path section.



Answer to the question no. 2

After successfully installation my Anaconda promt become:

i) Run simple python command and see the output:

```
Anaconda Prompt - pyspark

>>> list = [1, 22, 2, 3, 4]

>>> print(sorted(list))

[1, 2, 3, 4, 22]
```

ii) Run pi.py:

```
C:\Spark\spark-3.0.3-bin-hadoop2.7\examples\src\main\python>pyspark < pi.py
22/01/23 19:28:46 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Pi is roughly 3.141040

C:\Spark\spark-3.0.3-bin-hadoop2.7\examples\src\main\python>SUCCESS: The process with PID 20168 (child process of PID 18
392) has been terminated.
SUCCESS: The process with PID 18392 (child process of PID 14916) has been terminated.
SUCCESS: The process with PID 14916 (child process of PID 21900) has been terminated.
```

iii) spark-submit --master local[4] pi.py 2>session.log:

```
C:\Spark\spark-3.0.3-bin-hadoop2.7\examples\src\main\python>spark-submit --master local[4] pi.py 2>session.log
Pi is roughly 3.146460
C:\Spark\spark-3.0.3-bin-hadoop2.7\examples\src\main\python>
```

Answer to the question no. 3

```
In [2]: from __future__ import print_function
                                     import sys
                                      from random import random
                                     from operator import add
                                     from pyspark.sql import SparkSession
In [3]: if __name__ == "__main__":
                                                       spark = SparkSession\
                                                                       .builder\
.appName("PythonPi")\
                                                                          .getOrCreate()
                                                       slices = 1 \#int(sys.argv[1]) if len(sys.argv) > 1 else 2 n = 100000 * slices
                                                       def f(_):
                                                                        x = random() * 2 - 1
y = random() * 2 - 1
return 1 if x ** 2 + y ** 2 <= 1 else 0</pre>
                                                      count = spark.sparkContext.parallelize(range(1, n + 1), slices).map(f).reduce(add) print("Pi is roughly %f" % (4.0 * count / n))
                                                        spark.stop()
                                      \verb|C:\ProgramData\Anaconda3\lib\site-packages\pyspark\context.py: 238: Future \verb|Warning: Python 3.6 support is deprecated in Spark 3.6 support in Spark 3.6 support is deprecated in Spark 3.6 support 3.6 suppor
                                     Pi is roughly 3.144640
```