Spark installation on Windows

What you need to do for project 2:

1. Install
    - Java JDK 17
    - SBT
    - Apache Spark
    - hadoop.dll
    - winutils.exe
    - Install vc redist x64
2. Setup
3. Edit Scala files
4. Create a jar
5. Test your code

Installation:

1. Check if you have Java installed.
    - The version we will use requires Java 17, so please make sure you install that
    - If you already have Java JDK installed (check with java -version), then proceed to step 2.
    - If you do not have Java JDK installed, then follow the instructions in the link below
        - Amazon Corretto 17
        - Windows JDK Download link
2. Install sbt:
    - On your browser, search “Download sbt for Windows”
    - Click on the link “Download - SBT”
    - Select the SBT version mentioned in the overview documents (i.e sbt-1.9.4) and download the .msi file

sbt Learn Download Get Involved

Previous releases

1.x

sbt version	.zip	.zip.sha256	.tgz	.tgz.sha256	.msi	.msi.sha256
sbt 1.9.9	.zip	.zip.sha256	.tgz	.tgz.sha256	.msi	.msi.sha256
sbt 1.9.8	.zip	.zip.sha256	.tgz	.tgz.sha256	.msi	.msi.sha256
sbt 1.9.7	.zip	.zip.sha256	.tgz	.tgz.sha256	.msi	.msi.sha256
sbt 1.9.6	.zip	.zip.sha256	.tgz	.tgz.sha256	.msi	.msi.sha256
sbt 1.9.5	.zip	.zip.sha256	.tgz	.tgz.sha256	.msi	.msi.sha256
sbt 1.9.4	.zip	.zip.sha256	.tgz	.tgz.sha256	.msi	.msi.sha256
sbt 1.9.3	.zip	.zip.sha256	.tgz	.tgz.sha256	.msi	.msi.sha256
sbt 1.9.2	.zip	.zip.sha256	.tgz	.tgz.sha256	.msi	.msi.sha256
sbt 1.9.1	.zip	.zip.sha256	.tgz	.tgz.sha256	.msi	.msi.sha256

- Done!

3. Install Apache Spark:
   - On your browser, search “Download Apache Spark for Windows”
   - Click on the link “Downloads | Apache Spark ”
   - Scroll down to “Archived Releases” to get the version required
     - Click on the “spark release archives”
     - Select the version mentioned in the Overview Document (i.e Spark 3.4.1) and download the package highlighted below

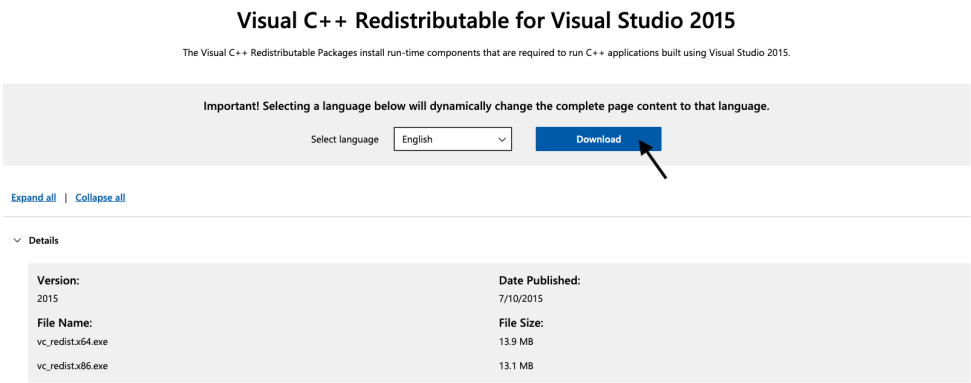
Index of /dist/spark/spark-3.4.1

Name	Last modified	Size	Description
Parent Directory		-	
SparkR_3.4.1.tar.gz	2023-06-19 23:25	348K	
SparkR_3.4.1.tar.gz.asc	2023-06-19 23:25	862	
SparkR_3.4.1.tar.gz.sha512	2023-06-19 23:25	150	
pyspark-3.4.1.tar.gz	2023-06-19 23:25	296M	
pyspark-3.4.1.tar.gz.asc	2023-06-19 23:25	862	
pyspark-3.4.1.tar.gz.sha512	2023-06-19 23:25	151	
spark-3.4.1-bin-hadoop3-scala2.13.tgz	2023-06-19 23:25	379M	
spark-3.4.1-bin-hadoop3-scala2.13.tgz.asc	2023-06-19 23:25	862	
spark-3.4.1-bin-hadoop3-scala2.13.tgz.sha512	2023-06-19 23:25	168	
spark-3.4.1-bin-hadoop3.tgz	2023-06-19 23:25	370M	
spark-3.4.1-bin-hadoop3.tgz.asc	2023-06-19 23:25	862	
spark-3.4.1-bin-hadoop3.tgz.sha512	2023-06-19 23:25	158	
spark-3.4.1-bin-without-hadoop.tgz	2023-06-19 23:25	286M	

   - After downloading, unzip the file
   - Copy-Paste the files and folders:
     - In C-drive → spark-local → spark
     - Create another folder in C-drive → spark-local → hadoop → bin

- From the repository (repo link):
  - Open: hadoop.dll
    1. Download and paste it into hadoop → bin
    2. **It also needs to be pasted in another folder: Root drive (C) → windows**
  - Open: winutils.exe
    1. Download and paste it into hadoop → bin

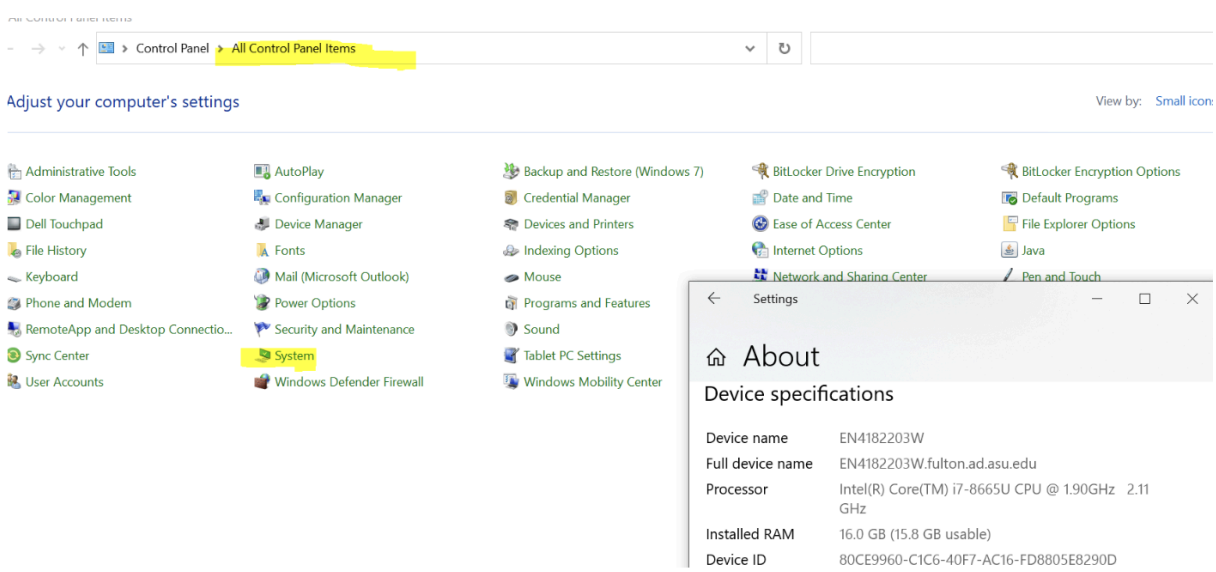
4. Install VC Redist x64
   - Go to the following page on your browser: https://www.microsoft.com/en-us/download/developer-tools
   - Scroll down and select Visual C++ Redistributable for Visual Studio 2015
   - Download vc_redist.x64.exe file



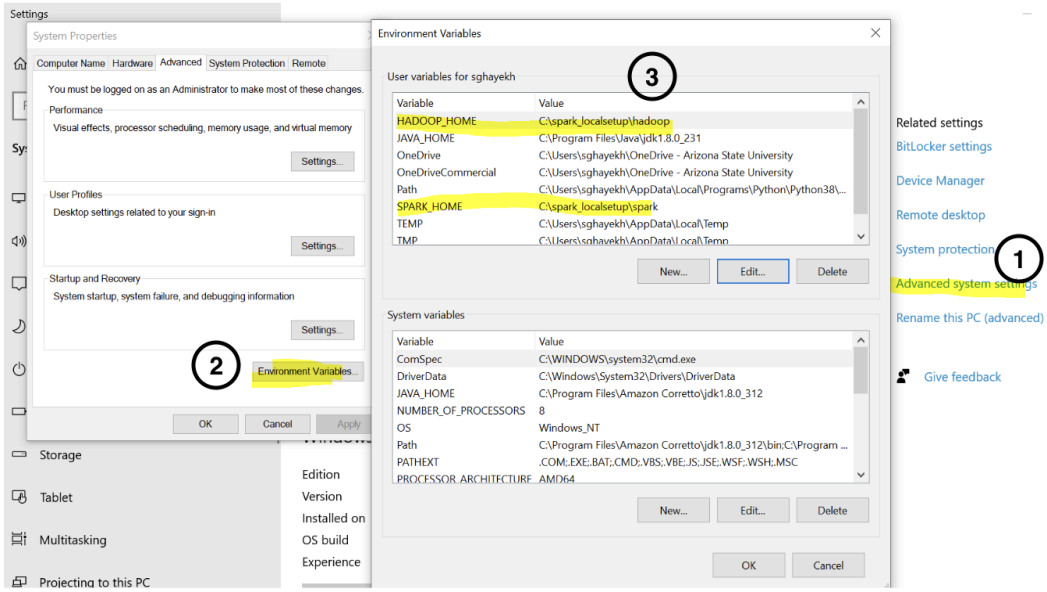
   - Done!!

Setup:

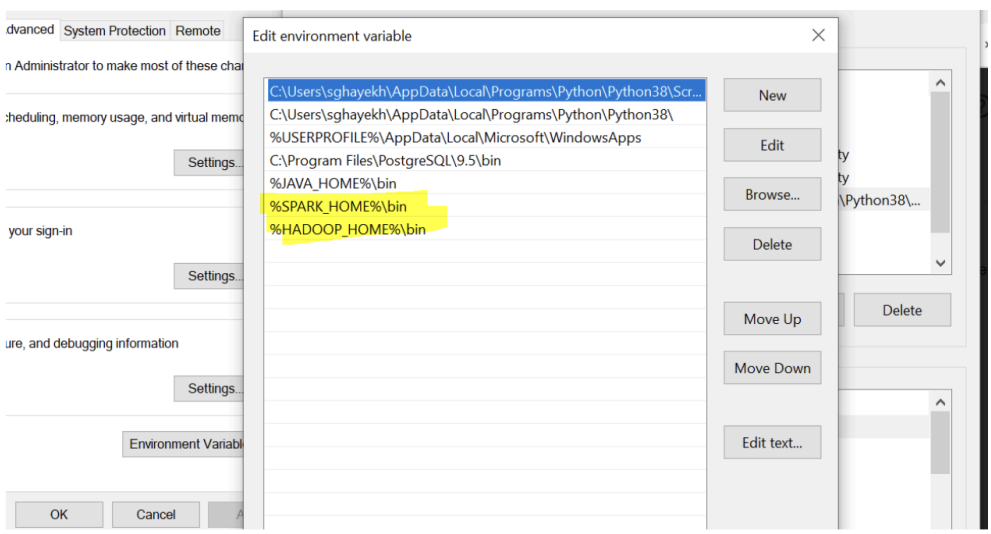
1. Go to Control Panel → All Control Panel Items → System



2. Open Advanced system settings → Environment Variables → Create two folders HADOOP_HOME and SPARK_HOME with their address in drive C.



3. Go to path:
   - Create 2 new lines to access their bin folder
     - %SPARK_HOME%\bin
     - %HADOOP_HOME%\bin



4. Restart your computer
5. After restarting:
   - Check JAVA: Open the command line to see if Java is working or not.

```
C:\Users\sghayekh>java
Usage: java [-options] class [args...]
           (to execute a class)
   or  java [-options] -jar jarfile [args...]
           (to execute a jar file)
where options include:
    -d32          use a 32-bit data model if available
    -d64          use a 64-bit data model if available
    -server       to select the "server" VM
                  The default VM is server.

    -cp <class search path of directories and zip/jar files>
    -classpath <class search path of directories and zip/jar files>
                  A ; separated list of directories, JAR archives,
                  and ZIP archives to search for class files.
    -D<name>=<value>
                  set a system property
    -verbose:[class|gc|jni]
                  enable verbose output
    -version      print product version and exit
    -version:<value>
                  Warning: this feature is deprecated and will be removed
                  in a future release.
                  require the specified version to run
    -showversion  print product version and continue
```

- Check SPARK: Open the command line and do the following commands
  - cd C:\<spark local setup>\spark
  - cd bin
  - spark-shell

```
C:\Windows\System32\cmd.e

Microsoft Windows [Version 10.0.22621.1928]
(c) Microsoft Corporation. All rights reserved.

C:\SPARK\bin>spark-shell
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
23/07/02 20:56:44 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
Spark context Web UI available at http://MSS-LP:4040
Spark context available as 'sc' (master = local[*], app id = local-1688311607701).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.4.1
      /_/

Using Scala version 2.12.17 (Java HotSpot(TM) 64-Bit Server VM, Java 17.0.7)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

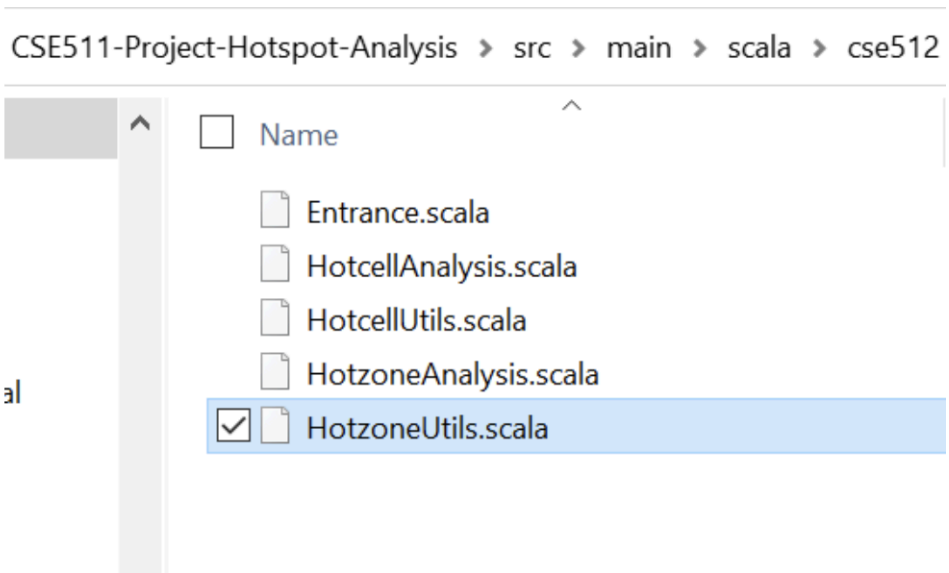
- Done !!

Edit Scala files:

1. Download the template: CSE 511_Hot Spot Analysis Project_Required Templates.zip
2. Extract it → then open

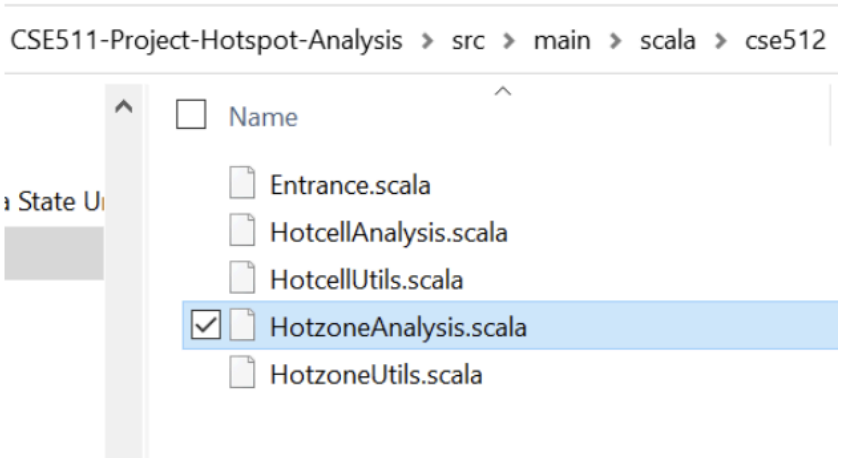
C:\yourpath\CSE511-Project-Hotspot-Analysis-20200804T194129Z-001\CSE511-Project-Hotspot-Analysis\src\main\scala\cse512

3. Hot zone analysis:
    - Input is a set of rectangles and set of points => aim finding the hotness zone/cell based on the number of points in each rectangle.
    - Function ST-Contains
        - Input 2 strings
            - Corner (opposite) points of the rectangle
            - point
    - Whether the point is in the rectangle or not?
    - You need to modify the below-highlighted file to write the function **ST-Contains.**




```
package cse512

object HotzoneUtils {

  def ST_Contains(queryRectangle: String, pointString: String ): Boolean = {
    // YOU NEED TO CHANGE THIS PART
    return true // YOU NEED TO CHANGE THIS PART
  }

  // YOU NEED TO CHANGE THIS PART

}
```

    - Then Open:

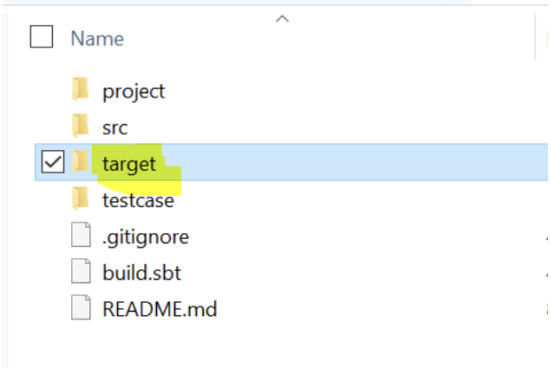



```
  // Parse point data formats
  spark.udf.register("trim",(string : String)=>(string.replace("(",
  "").replace(")", "")))
  pointDf = spark.sql("select trim(_c5) as _c5 from point")
  pointDf.createOrReplaceTempView("point")

  // Load rectangle data
  val rectangleDf =
  spark.read.format("com.databricks.spark.csv").option("delimiter","\t").option("h
  der","false").load(rectanglePath);
  rectangleDf.createOrReplaceTempView("rectangle")

  // Join two datasets
  spark.udf.register("ST_Contains",(queryRectangle:String,
  pointString:String)=>(HotzoneUtils.ST_Contains(queryRectangle, pointString)))
  val joinDf = spark.sql("select rectangle._c0 as rectangle, point._c5 as point
  from rectangle,point where ST_Contains(rectangle._c0,point._c5)")
  joinDf.createOrReplaceTempView("joinResult")

  // YOU NEED TO CHANGE THIS PART

  return joinDf // YOU NEED TO CHANGE THIS PART
}
```

- You need to add .coalesce(1) to the last query and this function merges all the partitions into a single partition and returns the output

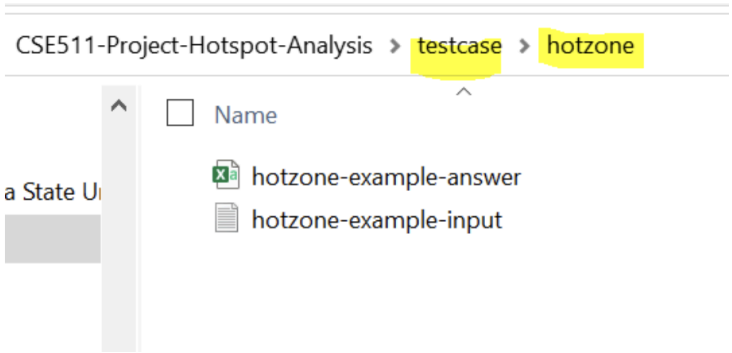
Create a jar file:

1. Once you finish your function in Scala, you need to create the jar file and then test it.
2. Go to the main root of the template:
   - Open the command line:
     - cd <your path to the project folder>
     - sbt assembly (takes some time to create a jar file)
     - You will find the jar file inside the target folder



Test your code

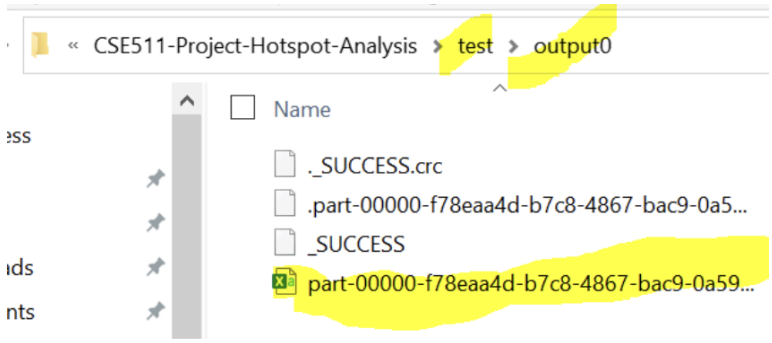
1. After creating the jar file, you need to test it.
2. Input test cases will be in the following folder: <your template folder path>\testcase\hotzone



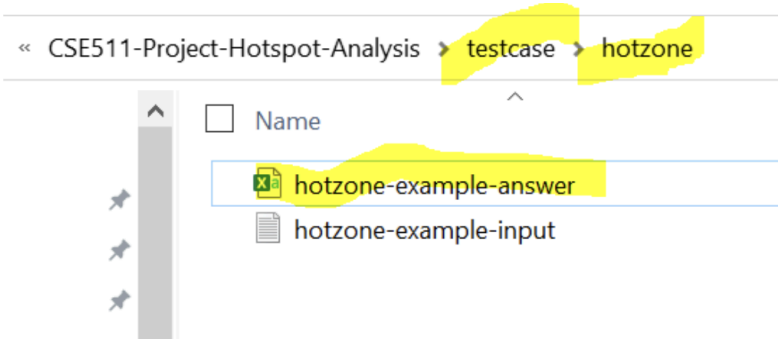
3. You need to write the following:

```
spark-submit
target\scala-<version>\CSE512-Hotspot-Analysis-Template-assembly-0.1.
0.jar test\output hotzoneanalysis src\resources\point-hotzone.csv
src\resources\zone-hotzone.csv
```

4. If everything is correct, you will see the output in the below path.



5. Its content must be the same as the output (hot zone-example-answer) as we have in the template:



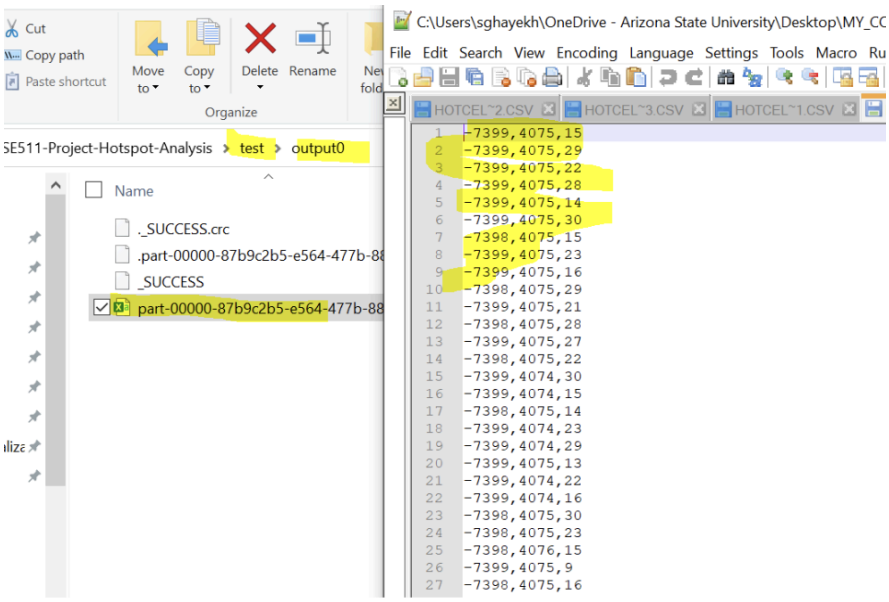
6. Done!!

For Hot Cell Analysis:

1. After applying changes in "**HotcellAnalysis.scala** and **HotcellUtils.scala**"
2. Then make a jar file using **sbt assembly**
3. Before testing the jar file,
    a. We also need to download the point data
        i. Download CSE 511_Hot Spot Analysis Project_yellow_trip_sample_100000.zip
        ii. Unzip and extract CSE 511_Hot Spot Analysis Project_yellow_trip_sample_100000.csv file.
    b. Paste the .csv file in the following path: <your project template folder path>\src\resources
4. The test input data will be in the following path: <your project template folder path>\testcase\hotcell\hotcell-example-input
5. You need to write the following command

```
spark-submit target\scala-<version>\CSE512-Hotspot-Analysis-Template-assembly-0.1.0.jar test\output hotcellanalysis src\resources\yellow_trip_sample_100000.csv
```

6. If everything is correct, you will see the output in the below path.


Cut
Copy path
Paste shortcut
Move to
Copy to
Delete
Rename
Organize
SE511-Project-Hotspot-Analysis > test > output0
Name
._SUCCESS.crc
.part-00000-87b9c2b5-e564-477b-88
_SUCCESS
part-00000-87b9c2b5-e564-477b-88
C:\Users\sghayekh\OneDrive - Arizona State University\Desktop\MY_CC
File Edit Search View Encoding Language Settings Tools Macro Ru
HOTCEL~2.CSV
HOTCEL~3.CSV
HOTCEL~1.CSV
-7399,4075,15
-7399,4075,29
-7399,4075,22
-7399,4075,28
-7399,4075,14
-7399,4075,30
-7398,4075,15
-7399,4075,23
-7399,4075,16
-7398,4075,29
-7399,4075,21
-7398,4075,28
-7399,4075,27
-7398,4075,22
-7399,4074,30
-7399,4074,15
-7398,4075,14
-7399,4074,23
-7399,4074,29
-7399,4075,13
-7399,4074,22
-7399,4074,16
-7398,4075,30
-7398,4075,23
-7398,4076,15
-7399,4075,9
-7398,4075,16

7. Done