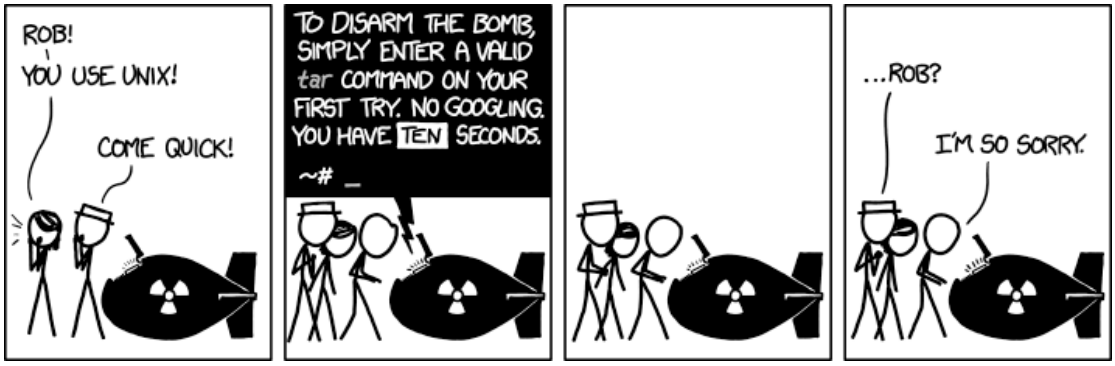
Student Guide for CS4225/CS5425 Assignment 2

1 Overview

1.1 Task



A team of engineers has planted a bomb somewhere in NUS. To disarm it, you need to log in to their network, which requires (1) the password of the bomb engineers' lab, and (2) the IP address of their engineering server.

To help you find this password and IP address, you have gained access to some log files that one of the engineers carelessly left in a public directory. Using these log files, with the help of some "notes to self" by one of the engineers, you need to search for the required information.

This is a new assignment, so feel free to let me know if you have any feedback, I'm happy to reply and to try to improve the assignment for future semesters ☺

1.2 Getting Started

To simplify the overall process compared to assignment 1, this assignment does not require using the school cluster at all (but it is still an option). In this assignment package, you will find **login.py**, which is a python script for checking your answers – it will ask you for the password and IP address, and helps you to check whether they are correct.

You should also find 2 folders, **part1** and **part2**. Each of these contains (1) a **NOTES.txt** file which gives you hints on the task you need to do, and (2) a log dataset, which you will run some queries on.

You can start by reading the **NOTES.txt** and scanning through the log files to get an idea of what you should do. There are 3 ways to run Spark: you can either (a) use

Google Colab (easy setup and allows editing code through notebook interface), (b) run Spark on the cluster (easy setup but slightly cumbersome to edit code, though fine if just using interactive shell) or (c) run Spark locally (more cumbersome setup). If you have no strong preference, I recommend (a), otherwise (b), otherwise (c).

You can use Spark in a language of your choice (Python, Scala, Java, R). It is fine to occasionally use non-Spark commands or packages (e.g. **pandas**) but please keep in mind that this assignment is designed to help you practice Spark, so try to do things in Spark as much as possible (note that Spark allows you to mix your code with other packages smoothly); and the assignment is designed to be smoothly solvable using Spark.

Also, keeping in mind the educational objectives of this assignment, please use Spark and not use 'brute force' to solve the assignment, i.e. testing a large number of possibilities using the script provided (**login.py**), or manually looking through a large number of lines of the log file. For both parts 1 and 2, **the correct solution should eliminate all but one possibility**, so no brute force should be needed. Note that the submission stage will expect you to explain the approach your solution uses.

2 Using Google Colab

To get started with Google Colab: visit the Colab notebooks:

- Part 1: https://colab.research.google.com/drive/1Vfp0Zm--TM3M_aqt-wrBZIZSg_OONA_U?usp=sharing
- Part 2: https://colab.research.google.com/drive/1CpS7cdYkB_UHFsqExk0Ka72Oncz-0r0V?usp=sharing

Check the comments in the Colab cells for tips on using Colab. To copy the notebooks so you can edit them, go to File > Save a copy in Drive. The notebooks have code set up for you for downloading and loading the datasets.

After running your Spark code in Google Colab, ideally you should have figured out the password (for part 1) and the network IP address (for part 2). To check your answers and submit, skip to section 5 of this guide.

3 Running on Cluster (Optional)

To run Spark on the server, you can login to the server as before (see the Assignment 1 guide); then transfer your files to the server via **scp** (see the Assignment guide if needed for info on how to do this).

Then, for Spark, use either **spark-submit** (for submitting Scala / PySpark jobs), **spark-shell** (for Scala shell), or **pyspark** (for Python shell). To check your answers and submit, skip to section 5 of this guide.

4 Local Environment Setup (Optional)

The instructions below are based on the assumption that you have followed the local setup instructions in Assignment 1.

If this is not the case, if you are using Python, the easiest way may be to use pip / conda: see https://spark.apache.org/docs/latest/api/python/getting_started/install.html. Otherwise, follow the parts of the previous guide for installing Java, then add the JAVA_HOME environment variable as before. Installing IntelliJ IDEA is optional – you can also just run your code locally using either **spark-submit**, **spark-shell** or **pyspark** similar to in section 3.

4.1 Install Spark

1. Download spark 3.0.0 build for Hadoop 3.2 using this link.
2. Extract the downloaded tarball to some directory. We'll use `/opt/spark` as an example in the following instructions.
3. Add SPARK_HOME environment variable: (`/path/to` below must be changed to the location where you just extracted spark, e.g. `/opt/spark`)

```
$ echo 'export SPARK_HOME=/path/to/spark-3.0.0-bin-hadoop3.2
' >> ~/.bash_profile
$ source ~/.bash_profile
```

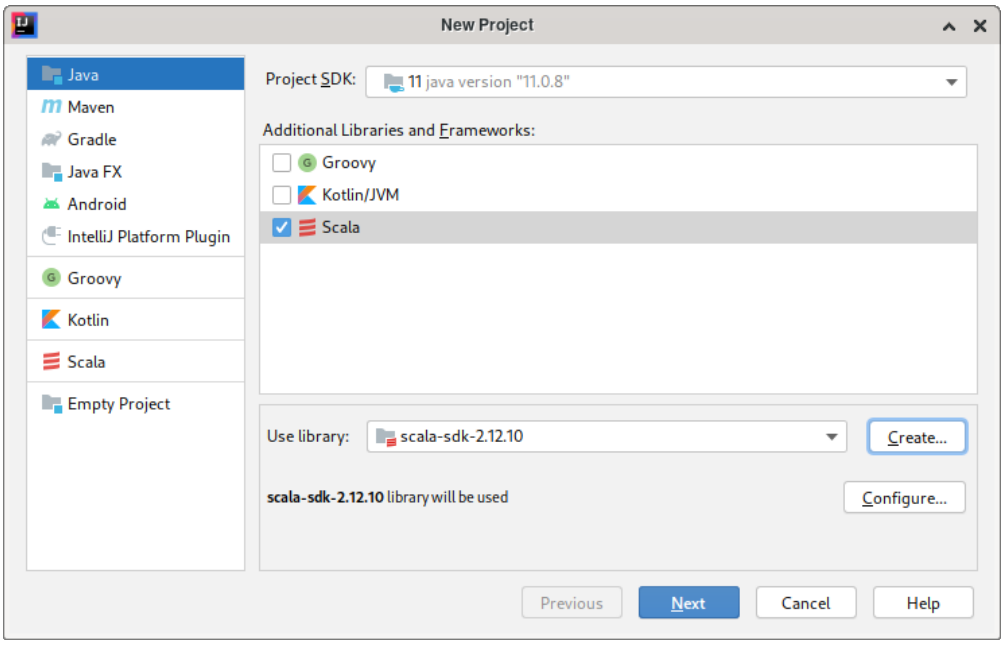
Or for Windows, follow the procedure in the previous guide (Windows section) to add this SPARK_HOME environment variable.

4.2 Install necessary components for IntelliJ IDEA

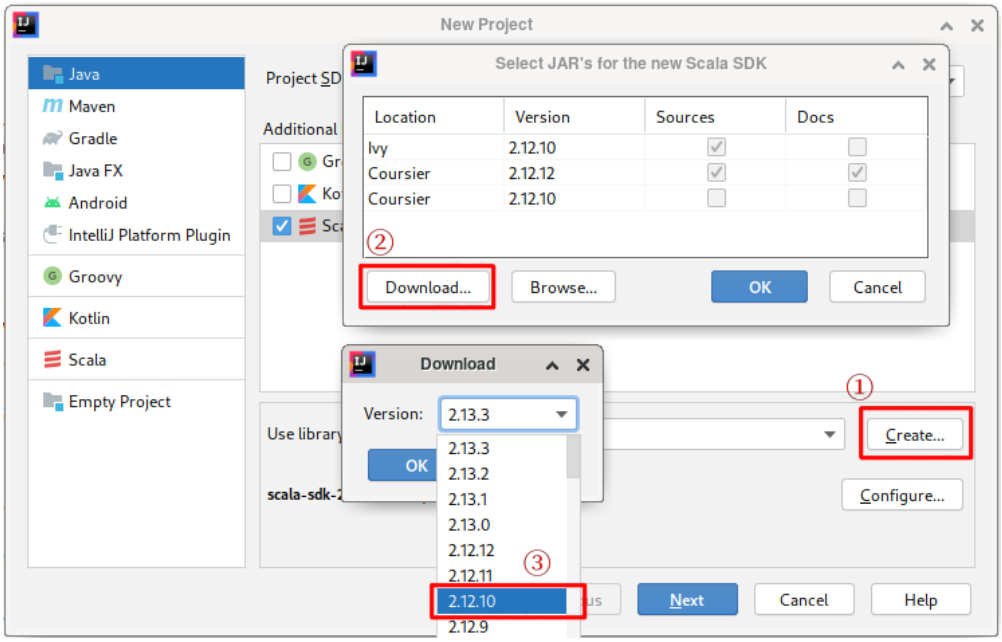
1. Open IntelliJ IDEA, navigate to `File > Settings > Plugins` and search for `Scala` and install the scala plugin.
2. After successful installation, restart IntelliJ IDEA.

4.3 Write scala in Intellij IDEA

1. Open IntelliJ IDEA, navigate to `File > New > Project`. On the left panel, select `Java`. On the right panel, tick `Scala`. In the `Use Library` dropdown list, select `scala 2.12.10`, as shown in the figure below.

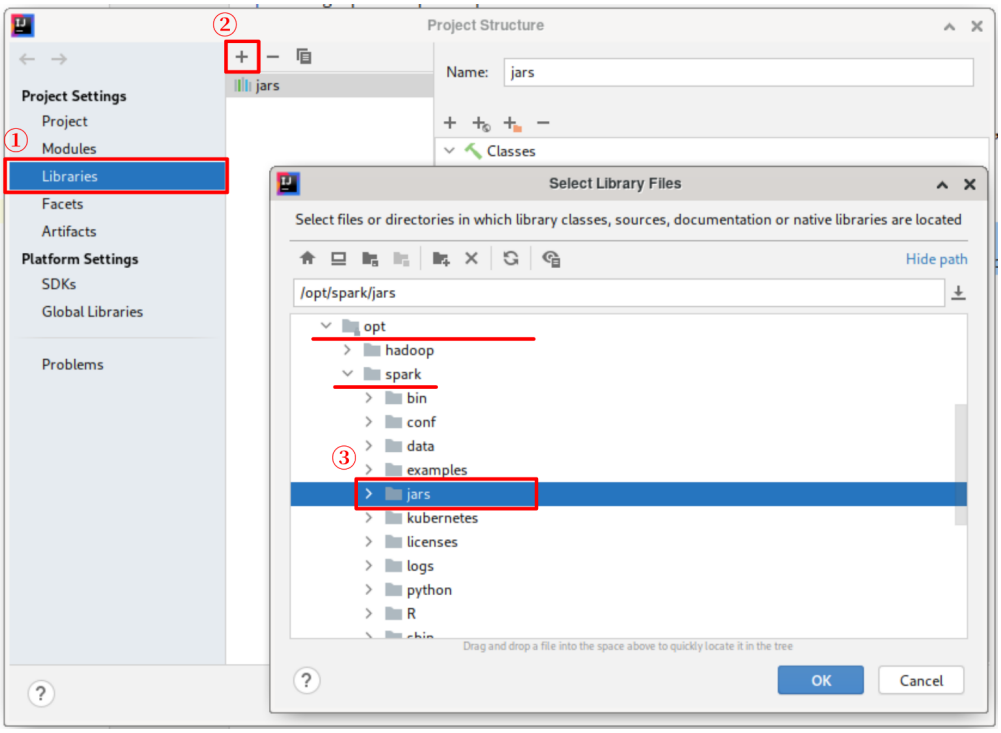


2. If it's not in the dropdown list, click the `Create...` button next to it, then click the `Download...` button in the popup window and download scala 2.12.10, as shown in the figure below. We will be using the same version of scala to test your code, so do not select other versions.



3. Click `Next`, choose the name and location for your project. Then click `Finish`.

4. Add spark libraries to your project. As shown below, navigate to `File > Project Settings > Libraries`, Click the "+" Button, select `Java`, then add `<spark-install-dir>/jars` to your project. In our case, it's `/opt/spark/jars`.



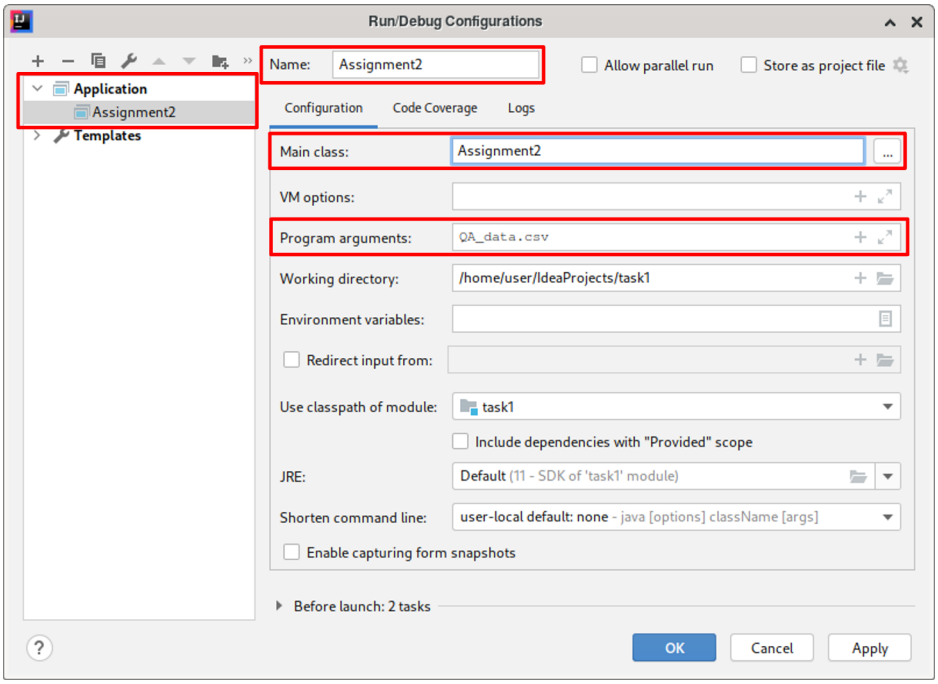
5. Add source file to your project. Right-click on the `src` directory in the project tree. Select `New > Scala Class`. In the popup window, select `Object` and name your scala object same as the name of the template file. Delete

everything in the main window. Copy the content of the template file and paste it into the main window. Then you are ready to start coding.

4.4 Running Your Code Locally

Once you have done editing the template, you will want to run and debug your source code. To do that, follow the instructions below:

1. Click `Add Configuration...` at the top-right corner of the IDE. In the pop-up window, click the "+" button and select `Application`.
2. Name your application, then select your main class (`Assignment2` for task 1, `Main` for task j2). Click `OK` and you are ready to run/debug your code. A successful configuration looks like the figure below.



5 Testing and Submitting

At this point, you should have the password (part 1) and/or IP address (part 2).

To **check your answers**, run **python login.py** in the assignment 2 directory (or **python3 login.py**, if necessary for running python 3). This script will ask you for the password and IP address; it will say "Success" if you are correct, or exit otherwise.

Once you are successful, please submit on LumiNUS. There are 2 things to submit:

1. Submit your answers on LumiNUS Quiz > Assignment 2. Copy and paste your

answers into the 2 blanks. For convenience, the answers you typed into **login.py** are also saved as a file called **answers.txt** in the same directory (but you don't have to submit answers.txt).

2. Copy and paste your code into submission.txt, and please provide a brief 1-2 sentence explanation describing your approach. Then, upload it on LumiNUS to the Assignment 2 submission folder. This submission will only be used as a "sanity check" just to ensure that you have a reasonable solution (and not e.g. obtained by brute-force trying passwords or extracting the answers from the **login.py** script). We will not consider things like efficiency / coding quality for grading – and we will not test the code on any other datasets – as long as your code and explanations are "sane" and your answers are correct, you will get full marks. I may look through to see what approaches students took, to help me improve the assignment for future.

6 Links and References

For a basic guide + API reference for Spark, see https://spark.apache.org/docs/latest/sql-getting-started.html, or https://spark.apache.org/docs/latest/api/python/getting_started/index.html for PySpark. For a PySpark 'cheatsheet', see https://s3.amazonaws.com/assets.datacamp.com/blog_assets/PySpark_SQL_Cheat_Sheet_Python.pdf.