# How to Create a Fair Grading Distribution for Quizzes/Exams in CMSC131/132: A Guide for New Lead Teaching Assistants

## Introduction

These instructions are for Lead Teaching Assistants for CMSC131 or CMSC132 with Pedram Sadeghian. These instructions will present a procedure for using the script 'grading_assign.py' to routinely create a grading distribution for a quiz or exam. This task can be done by hand, but it is tedious and time consuming, using the script will make the process much quicker and format the distribution nicely for quickly sending to the team. All names provided in the examples are made up to protect the privacy of the current TAs.

## Caution

These instructions involve the use of the command prompt or terminal commands. Assuming you are using a computer which you own, you will be acting with an increased privilege level in your operating system. Use only the commands given in the instructions, the author cannot guarantee that the usage of any alternative commands will produce the same results. Using commands you are not familiar with may result in damage to your computer or unintended deletion of files.

## Technical Background

1. As a Computer Science major, you should know how to create files in your file system.
    - You will need to be able to create .txt files and .csv files.
2. For these instructions, you will need to be able to navigate through your file system using the command prompt (for Windows) or terminal (MacOS/Linux) and execute Python programs from the command line.

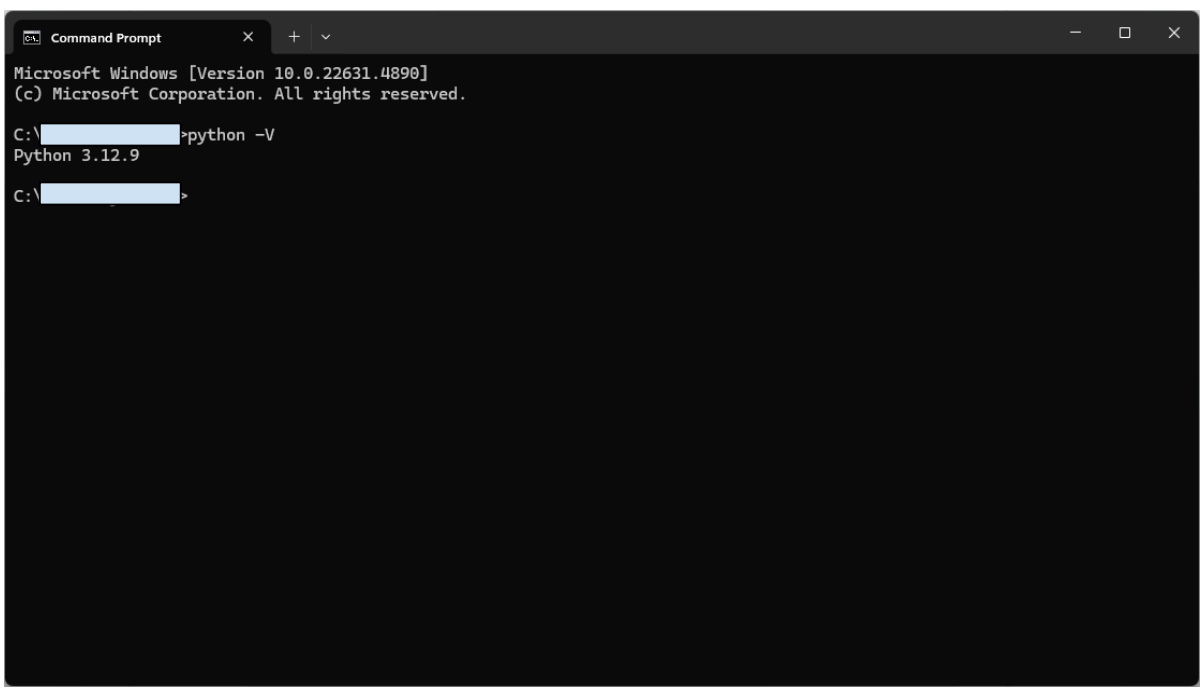
- If you do not know how to navigate the file system, refer to this (Thackston, R., 2020) video for Windows or this (Schafer, C., 2015) video for MacOS and Linux.
- If you do not know how to execute Python scripts from the command line, refer to the man pages here (Python Software Foundation, n.d.) for Windows and here (Python Software Foundation, n.d.-b) for MacOS/Linux.

3. As a Lead TA for CMSC131/132 with Pedram, Slack is the main form of communication used. Thus, you should be familiar with navigating channels, attaching files to messages, and pinging groups.

**Materials**

For these instructions you will require:

1. A computer with the capability to run Python
2. Python 3.12.9 or later installed
    a. To check your Python installation, open the terminal and type "**python -V**". The version should be printed to the standard output (see example below)

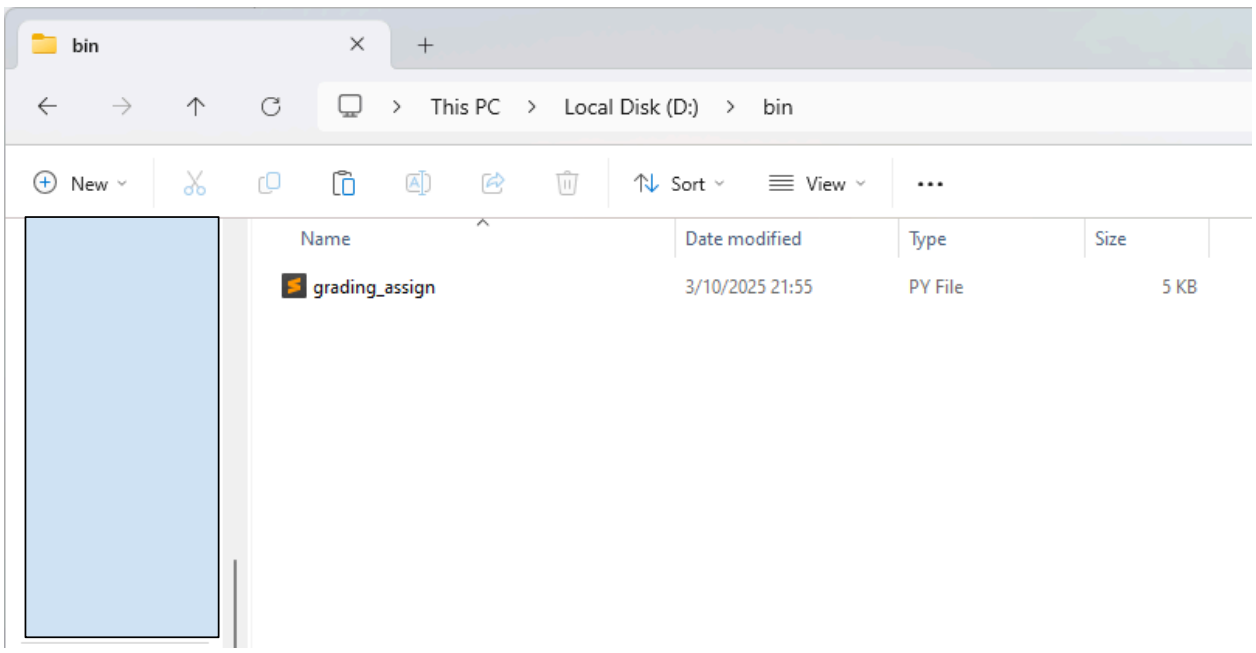


3. The python script "**grading_assign.py**" which should be provided to you by a senior lead.
4. Knowledge of the names of the TAs in your team

**Procedure**

**Part I: Creating the workspace**

1. Using your regular file explorer, navigate to the location where you want to store the scripts and related files.
2. Create a new folder and place the **grading_assign.py** script file in the folder.
    a. In this example, I have created a folder called **bin** to place the script in.




**Part II: Creating the file "tas.csv"**

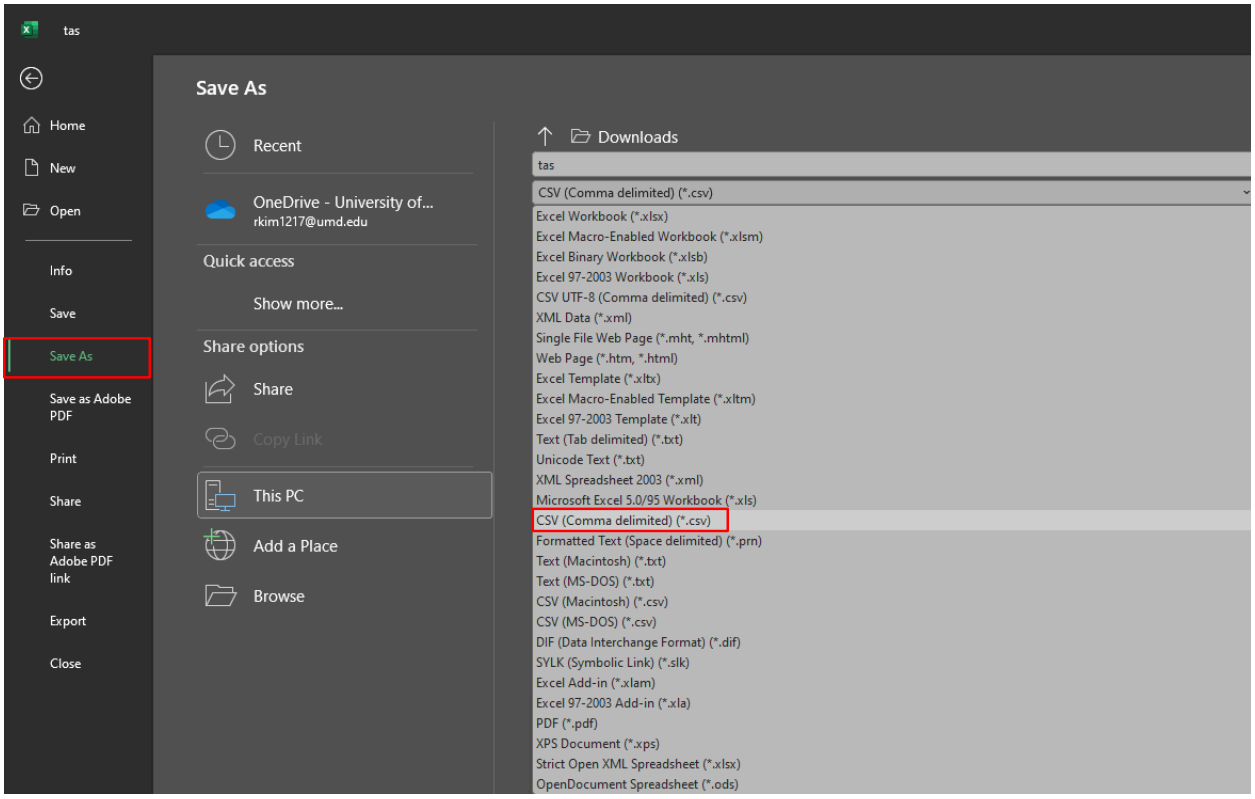
1. Open a new Excel file
2. In **column A**, write the names of all of the Teaching Assistants, each on its own row.
3. In **column B**, write the directoryIDs of the Teaching Assistant

4. In **column C**, type **1** if the TA is an undergraduate, and type a **2** if the TA is a graduate student.
5. In **column D**, type "**10 hours**" if the TA is an undergraduate, and type "**20 hours**" if the TA is a graduate student.
6. Check to make sure your sheet looks like the example below.
    a. In the example, John, Jill, Jack, Bill, and Bob are undergraduate students. Jane, Amy, Colleen, Varun, and Robert are graduate students.

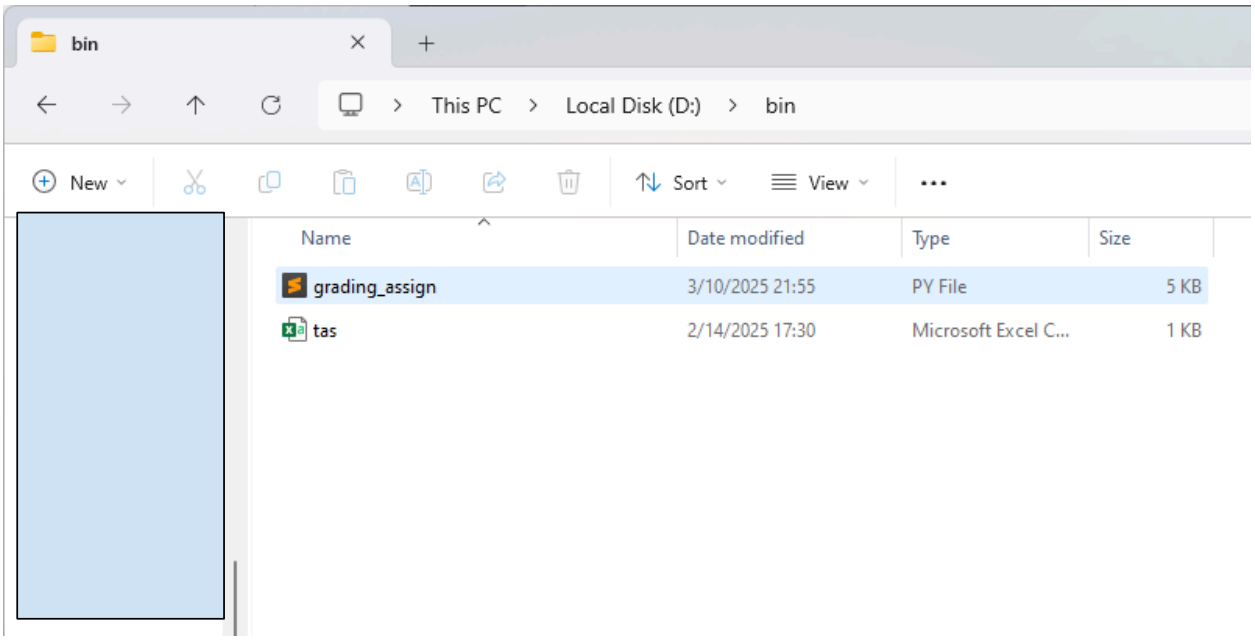
G18

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | John | johndoe | 1 | 10 hours | |
| 2 | Jane | janedoe123 | 2 | 20 hours | |
| 3 | Jill | jidoe | 1 | 10 hours | |
| 4 | Jack | mjack5 | 1 | 10 hours | |
| 5 | Bill | billyjoe | 1 | 10 hours | |
| 6 | Bob | bobertyor | 1 | 10 hours | |
| 7 | Amy | akarak8 | 2 | 20 hours | |
| 8 | Colleen | colju | 2 | 20 hours | |
| 9 | Varun | getvarun | 2 | 20 hours | |
| 10 | Robert | robw34 | 2 | 20 hours | |
| 11 | | | | | |

7. Save the file as **tas.csv** using the **Save as** option, ensuring that **CSV (Comma delimited) (*.csv)** is selected from the extension dropdown.




8. Save the file in the same folder that you placed the **grading_assign.py** script.



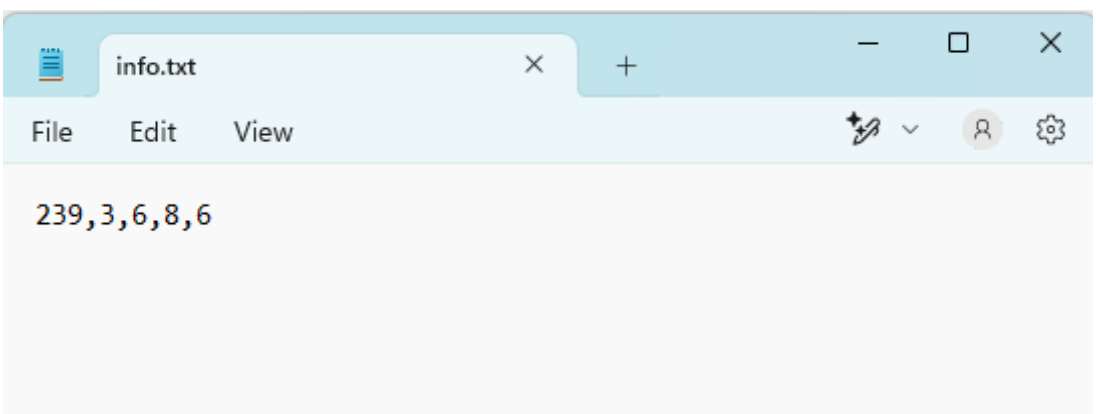

## Part III: Creating the input file for the script

1. Open a new txt file

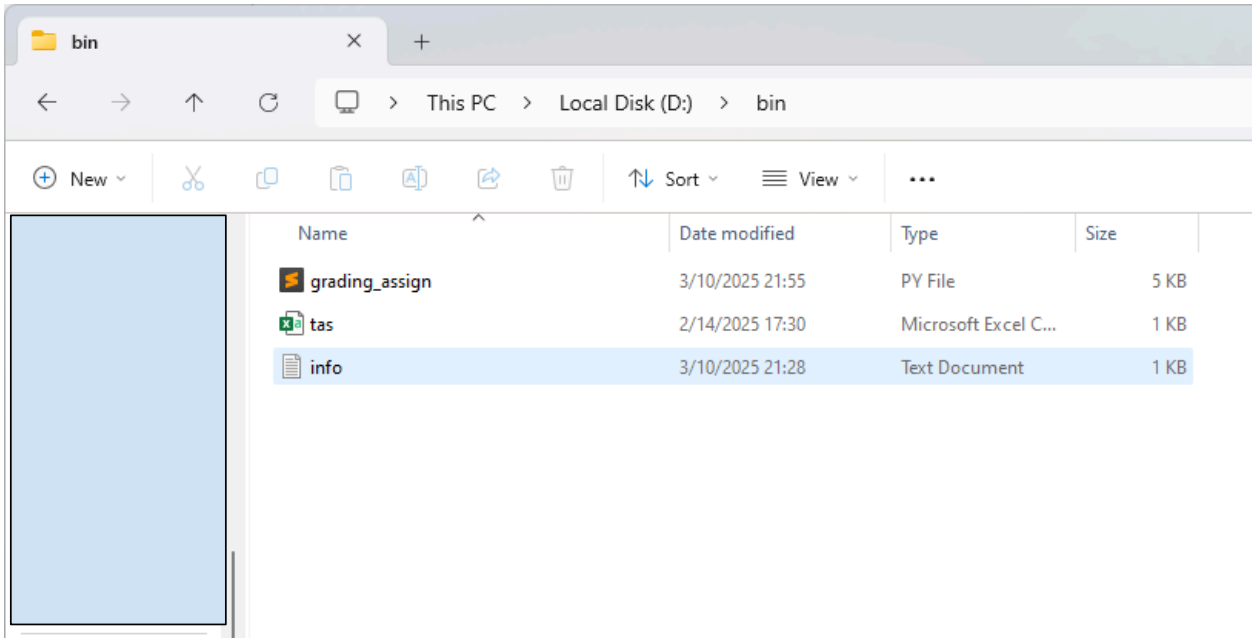
2. Type the **number of student submissions** followed by a comma, then the **number of questions which need to be graded**.
3. Decide upon a fair weighting for each question. This should represent approximately how much time it should take to grade 1 submission for each question.

   **Note** - If you are creating a distribution for a quiz, then the weighting does not matter. Simply consider it as taking 1 minute.
4. For each question, type the **weight** followed by a comma, except for the last question, where the comma should be omitted

   **Note** - If you are curious about how the weights are used, the script uses a system similar to stride scheduling to distribute the submissions. To learn more about the general idea, refer to the original paper on stride scheduling by Carl Waldspurger (1995, p. 48-55).
5. Check your text file against the example below
   a. In the example, there are **239** submissions and **3** questions to be graded. Question 1 takes **6** minutes, question 2 takes **8** minutes, and question 3 takes **6 minutes**.

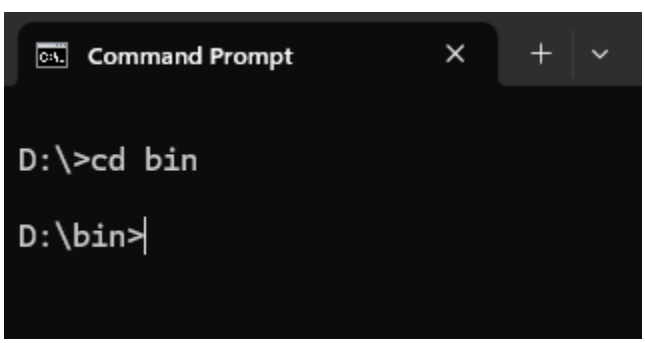



6. Save the txt file as **info.txt**. Save the file in the same folder that you placed the **grading_assign.py** script and the **tas.csv** sheet.




**Part IV: Running the script**

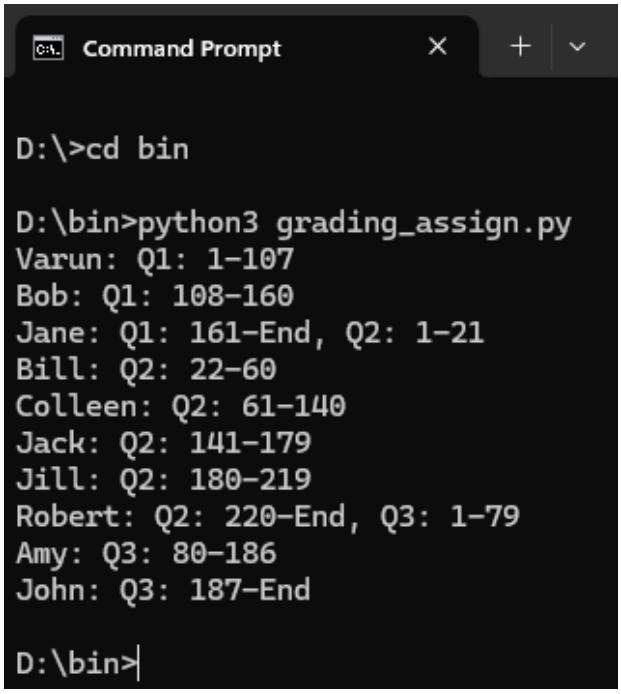
1. Open your terminal.
2. Navigate to the folder where you placed the **grading_assign.py** script in **Part I: Creating the workspace** using the **cd** command.
    a. In the example below, my scripts are located in the **bin** folder on my **D:** drive so I used the command **cd bin**




**Note** - You may have to switch drives if your operating system is installed on a different drive than where you saved the script. To do so, type the drive letter followed by a colon, then press Enter. For example, if the script is located on the D drive then type "**D:**".

3. Run the python script by typing the command "**python3 grading_assign.py**".

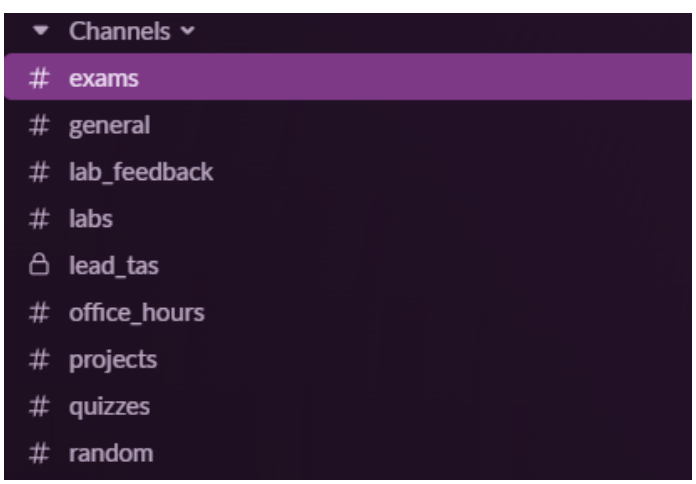
4. Your output should look something like the example below.



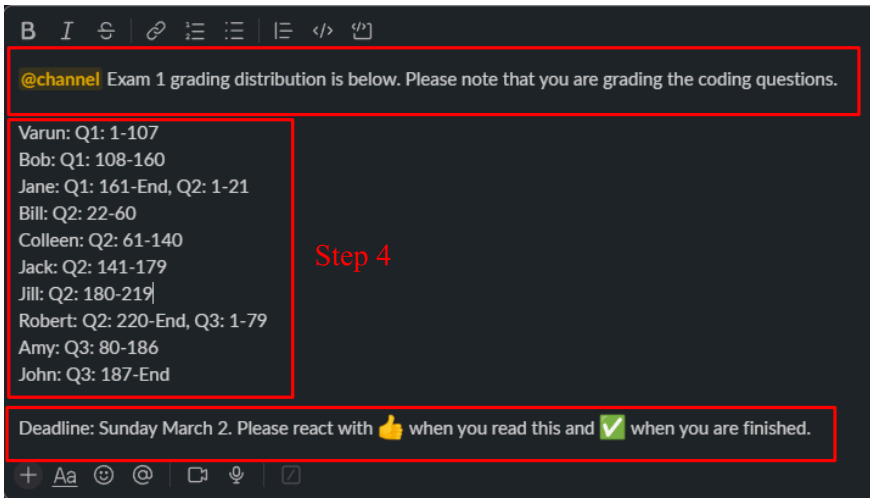

**Part V: Sending the Grading Distribution Out**

1. Open Slack on your desktop or through your browser

2. Navigate to the appropriate channel. For example, **#exams** if this grading distribution is for an exam



3. Ping the channel and type a message describing which assignment the distribution is for.
4. Copy the output from the **grading_assign.py** script and paste it below your message.
5. Include the grading deadline, and remind everyone to react to the message when they read your message and when they finish so you can track grading progress.



## References


Department of Computer Science. (n.d.). *Degree requirements for CS Major*. Degree Requirements for CS Major | Undergraduate Computer Science at UMD. https://undergrad.cs.umd.edu/degree-requirements-cs-major

Department of Computer Science. (n.d.). *Teaching assistants*. Undergraduate Computer Science at UMD. https://undergrad.cs.umd.edu/teaching-assistants

Knight, R. (2016, November 14). *Make sure your team's workload is divided fairly*. Harvard Business Review. https://hbr.org/2016/11/make-sure-your-teams-workload-is-divided-fairly

Python Software Foundation. (n.d.). *Python on windows FAQ*. Python documentation. https://docs.python.org/3/faq/windows.html#how-do-i-run-a-python-program-under-windows

Python Software Foundation. (n.d.-b). *Using Python on MacOS*. Python documentation. https://docs.python.org/3/using/mac.html#how-to-run-a-python-script

Schafer, C. (2015, October 21). *Linux/Mac Terminal Tutorial: Navigating your Filesystem*. YouTube. https://www.youtube.com/watch?v=j6vKLJxAKfw

Thackston, R. (2020, February 19). *[044] Windows Command Prompt - Navigating the file system*. YouTube. https://www.youtube.com/watch?v=9zMWXD-xoxc

Waldspurger, C. A. (1995). *Lottery and stride scheduling: Flexible proportional-share resource management* (thesis). *Lottery and stride scheduling: flexible proportional-share resource management*. Massachusetts Institute of Technology. Laboratory for Computer Science, Cambridge, Mass.