1 Python from the command line

If your python code is in the file `main.py` you can run it from the command line with:

```
$ python main.py
```

For the following python code:

main.py

```
import sys
print(sys.argv)
```

Running on the command line will produce the following:

```
$ python main.py 1 2 3
[’main.py’, ’1’, ’2’, ’3’]
```

2 Bash shell commands

The online tutorial we used is here: `http://www.ee.surrey.ac.uk/Teaching/Unix/`
Below is a partial cheat sheet from
`https://github.com/RehanSaeed/Bash-Cheat-Sheet/blob/main/README.md`
You should be familiar with the following unix/bash commands that we covered in class:

```
## Navigating Directories

pwd                          # Print current directory path
ls                           # List directories
ls -a|--all                  # List directories including hidden
ls -l                        # List directories in long form
ls -l -h|--human-readable    # List directories in long form with human readable sizes
ls -t                        # List directories by modification time, newest first
cd foo                       # Go to foo sub-directory
cd                           # Go to home directory
cd ˜                         # Go to home directory
cd -                         # Go to last directory

## Directories

mkdir foo                           # Create a directory
mv foo bar                          # Move directory
rmdir foo                           # Delete non-empty directory

## Standard Output, Standard Error and Standard Input
```

```
echo "foo" > bar.txt       # Overwrite file with content
echo "foo" >> bar.txt      # Append to file with content
```

## Moving Files

```
cp foo.txt bar.txt                               # Copy file
mv foo.txt bar.txt                               # Move file
```

## Deleting Files

```
rm foo.txt            # Delete file
```

## Reading Files

```
cat foo.txt            # Print all contents
less foo.txt           # Print some contents at a time
   (g - go to top of file, SHIFT+g, go to bottom of file, /foo to search for 'foo')
head foo.txt           # Print top 10 lines of file
tail foo.txt           # Print bottom 10 lines of file
open foo.txt           # Open file in the default editor
wc foo.txt             # List number of lines words and characters in the file
```

## File Permissions

#	Permission	rwx	Binary
7	read, write and execute	rwx	111
6	read and write	rw-	110
5	read and execute	r-x	101
4	read only	r--	100
3	write and execute	-wx	011
2	write only	-w-	010
1	execute only	--x	001
0	none	---	000

For a directory, execute means you can enter a directory.

- u - User
- g - Group
- o - Others
- a - All of the above

```
ls -l foo.sh            # List file permissions
```

```
chmod u+x foo.sh         # Give the user execute permission
chmod g+x foo.sh         # Give the group execute permission
chmod u-x,g-x foo.sh     # Take away the user and group execute permission
chmod u+x,g+x,o+x foo.sh # Give everybody execute permission
chmod a+x foo.sh         # Give everybody execute permission
chmod +x foo.sh          # Give everybody execute permission
```

## Finding Files

Find binary files for a command.

```
which wget                                 # Find the binary
```

## Find in Files

```
grep 'foo' bar.txt                         # Search for 'foo' in file 'bar.txt'
```

## Disk Usage

```
df                     # List disks, size, used and available space
du                     # List current directory, subdirectories and file sizes
```

## Identifying Processes

```
ps all                 # List all processes
CTRL+Z                 # Suspend a process running in the foreground
bg                     # Resume a suspended process and run in the background
fg                     # Bring the last background process to the foreground
fg 1                   # Bring the background process with the PID to the foreground

sleep 30 &             # Sleep for 30 seconds and move the process into the background
jobs                   # List all background jobs
jobs -p                # List all background jobs with their PID
```

## Killing Processes

```
CTRL+C                 # Kill a process running in the foreground
kill PID               # Shut down process by PID gracefully. Sends TERM signal.
kill -9 PID            # Force shut down of process by PID. Sends SIGKILL signal.
```