Lecture 7 – Code Workshop

Naveh Porat

June 2022

Learning Code

- A large portion of the course is dedicated to code.
- Today we will see some useful tools and coding concepts.
- It’s very hard to teach\learn code because it tends to be very technical.
- We’ll try to have everything demonstrated live.
- Try to follow along and ask as many questions as you can!

What is good code?

We would like to write code that :

1. Is easy to use
2. Is easy to read (by others)
3. Is easy to modify
4. Is fast

Table of Contents

Pycharm - Shortcuts

Text Editing

Ctrl + right/left

```
def foo():
    long_text = "This is a text containing many words"
    words = []
    for i in long_text.split(" "):
        words.append(i)
    return words
```

Home & End

```
def foo():
    long_text = "This is a text containing many words"
    words = []
    for i in long_text.split(" "):
        words.append(i)
    return words
```

Ctrl + w

```
def foo():
    long_text = "This is a text containing many words"
    words = []
    for i in long_text.split(" "):
        words.append(i)
    return words
```

Ctrl + c & Ctrl + v

```
def foo():
    long_text = "This is a text containing many words"
    words = []
    for i in long_text.split(" "):
        words.append(i)
    return words
```

Ctrl + x & Ctrl +v

```
def foo():
    long_text = "This is a text containing many words"
    words = []
    for i in long_text.split(" "):
        words.append(i)
    return words
```

Ctrl + d

```
def foo():
    long_text = "This is a text containing many words"
    words = []
    for i in long_text.split(" "):
        words.append(i)
    return words
```

Code Folding

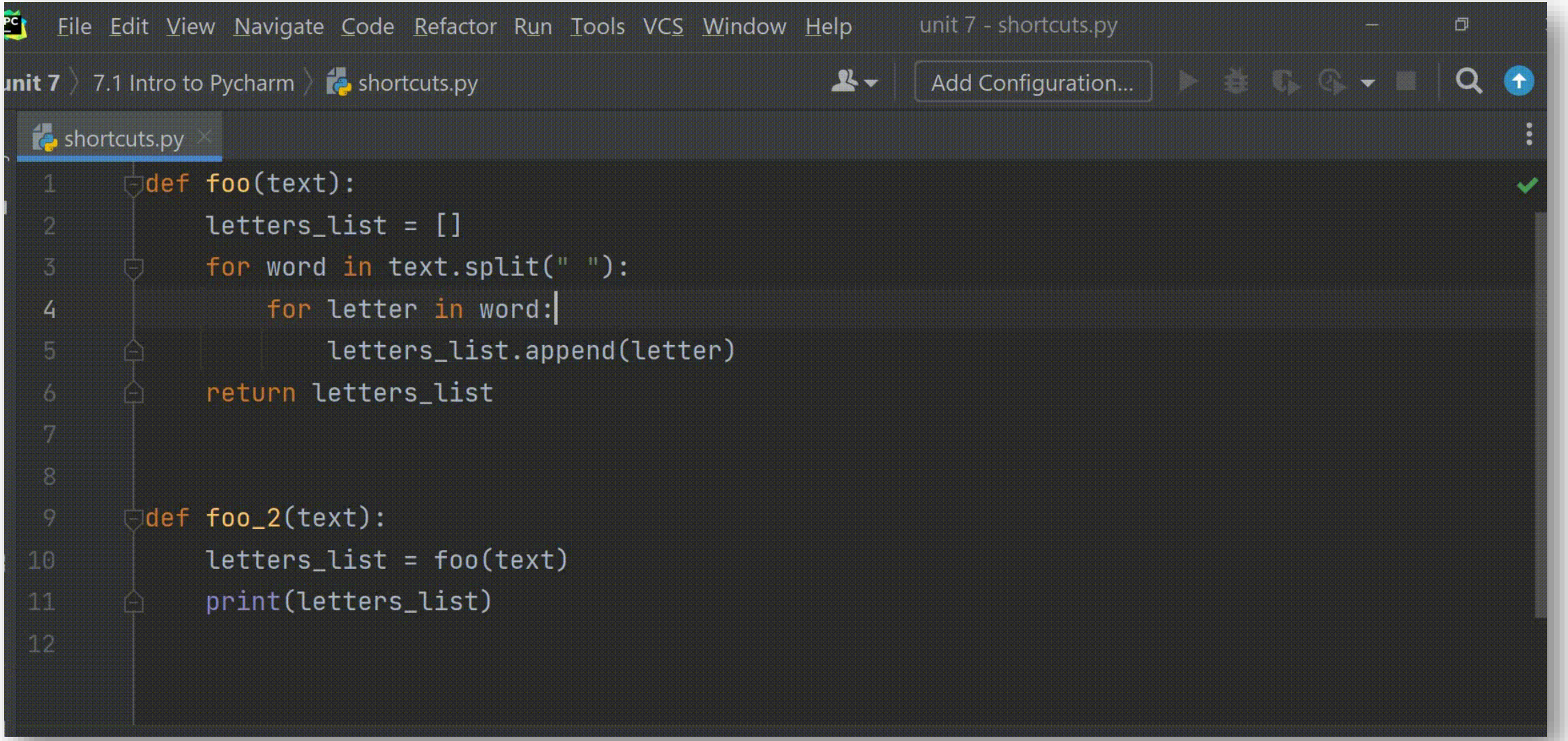
```
def foo(text):
    letters = []
    for word in text.split(" "):
        for letter in word:
            letters.append(letter)
    return letters


def foo_2(text):
    letters = foo(text)
    print(letters)
```

Commenting

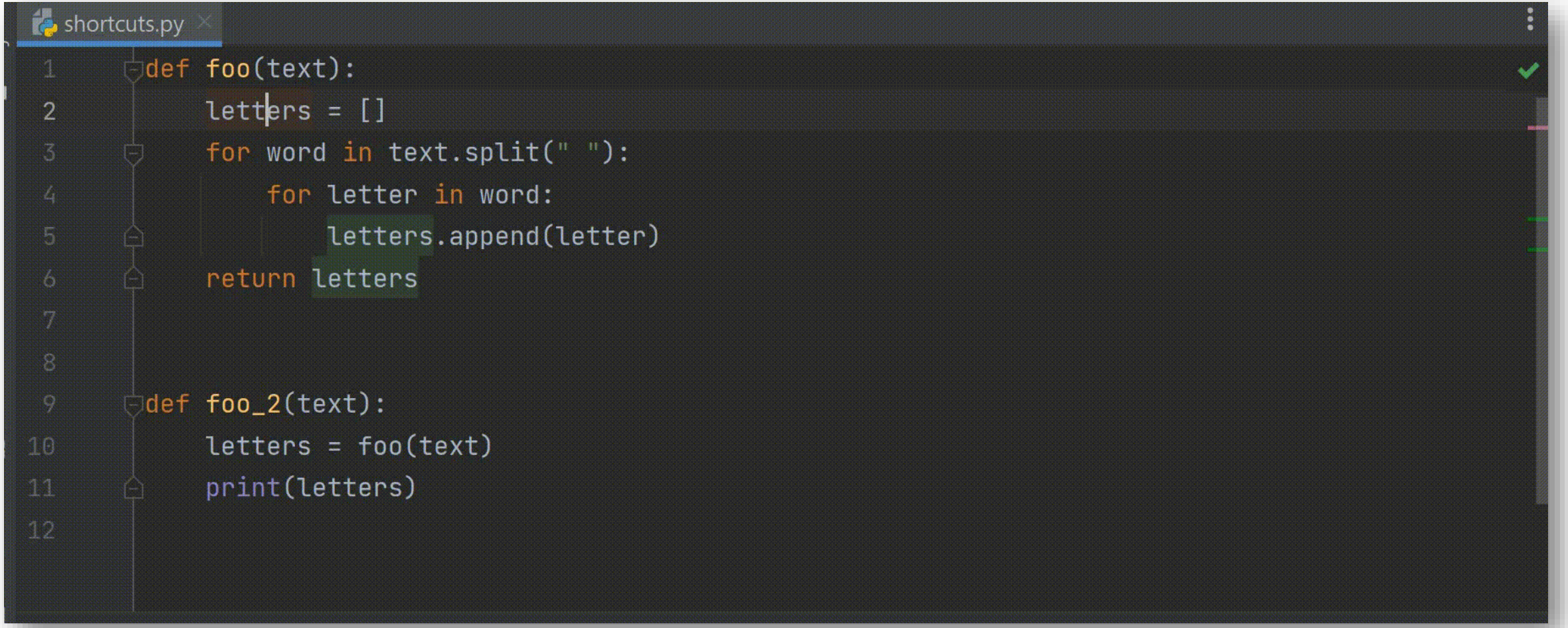
```
def foo():
    long_text = "This is a text containing many words"
    words = []
    for i in long_text.split(" "):
        words.append(i)
    return words
```

Search & Replace

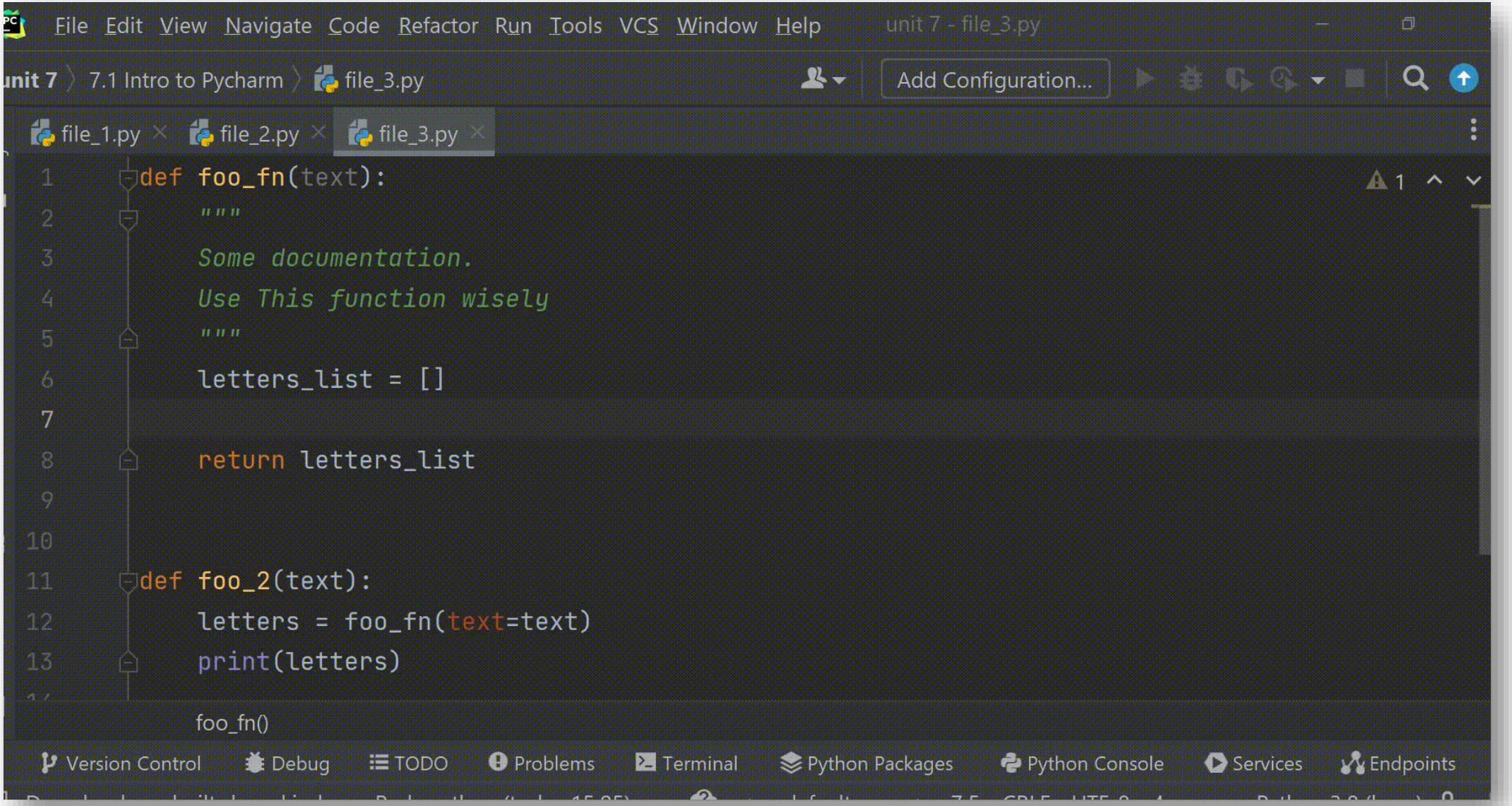
Global Search & Replace



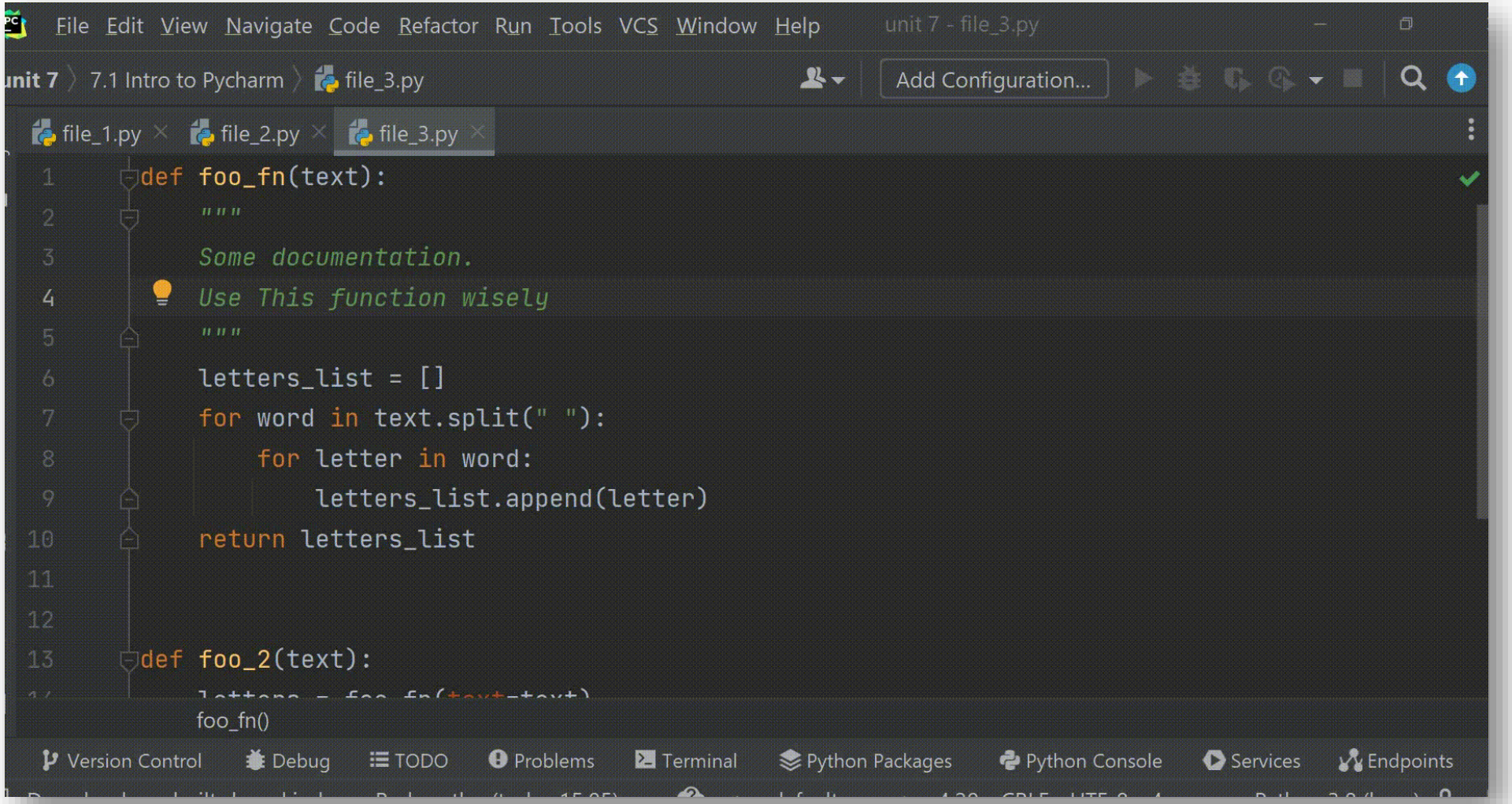
Rename


shortcuts.py
def foo(text):
letters = []
for word in text.split(" "):
for letter in word:
letters.append(letter)
return letters
def foo_2(text):
letters = foo(text)
print(letters)

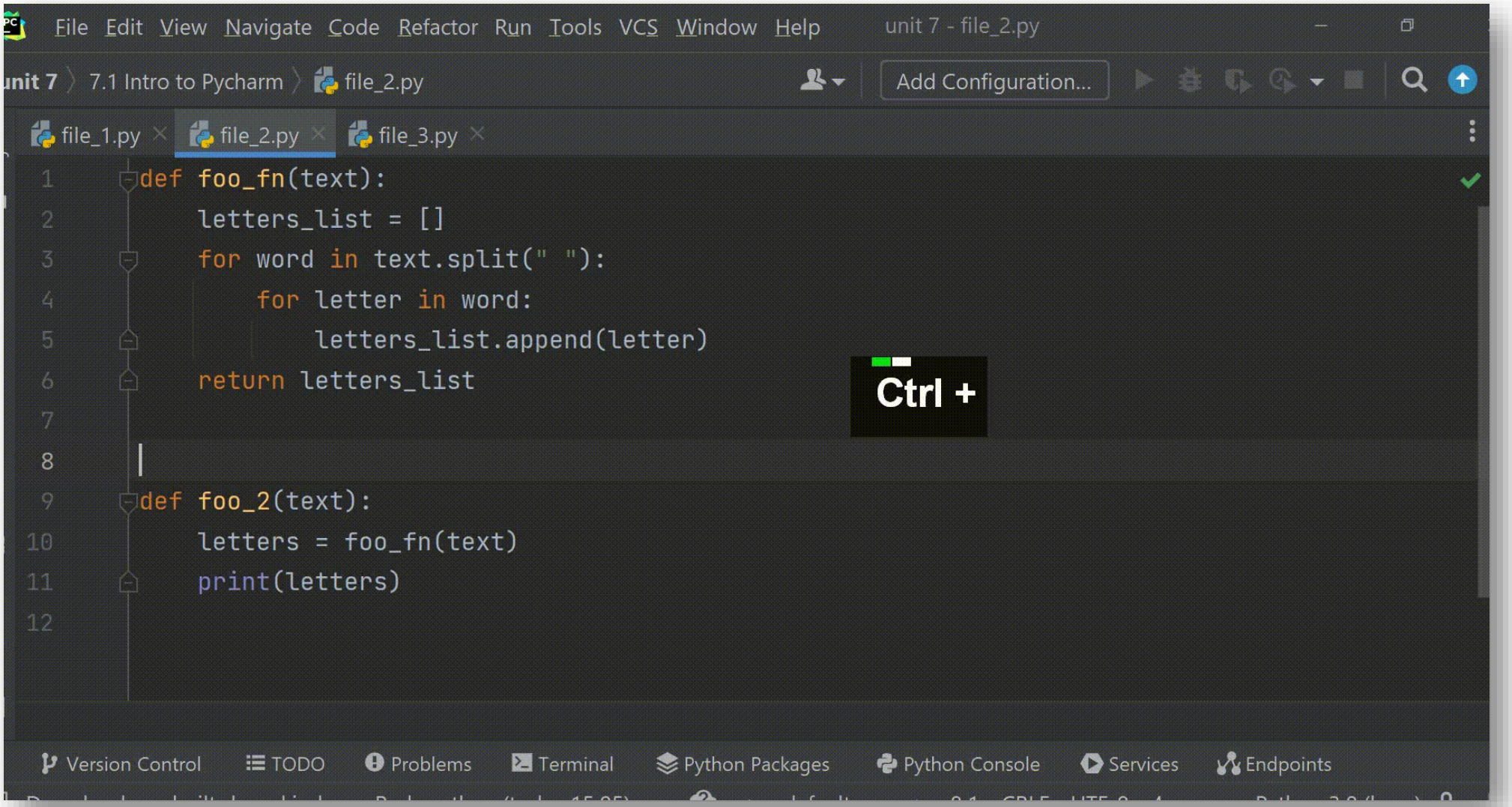
Smart Completion



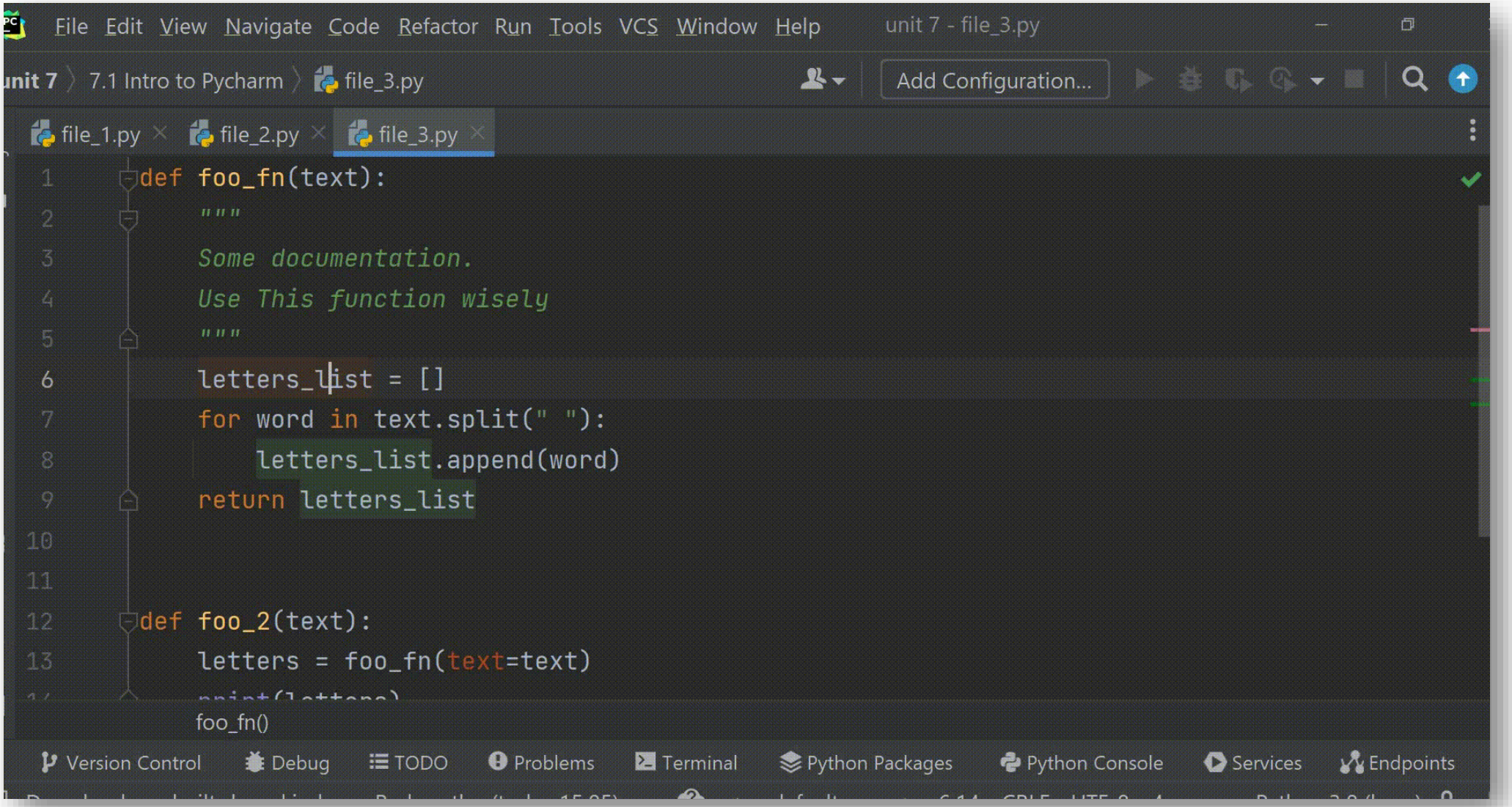
Documentation, Param & Referencing



Navigating Windows



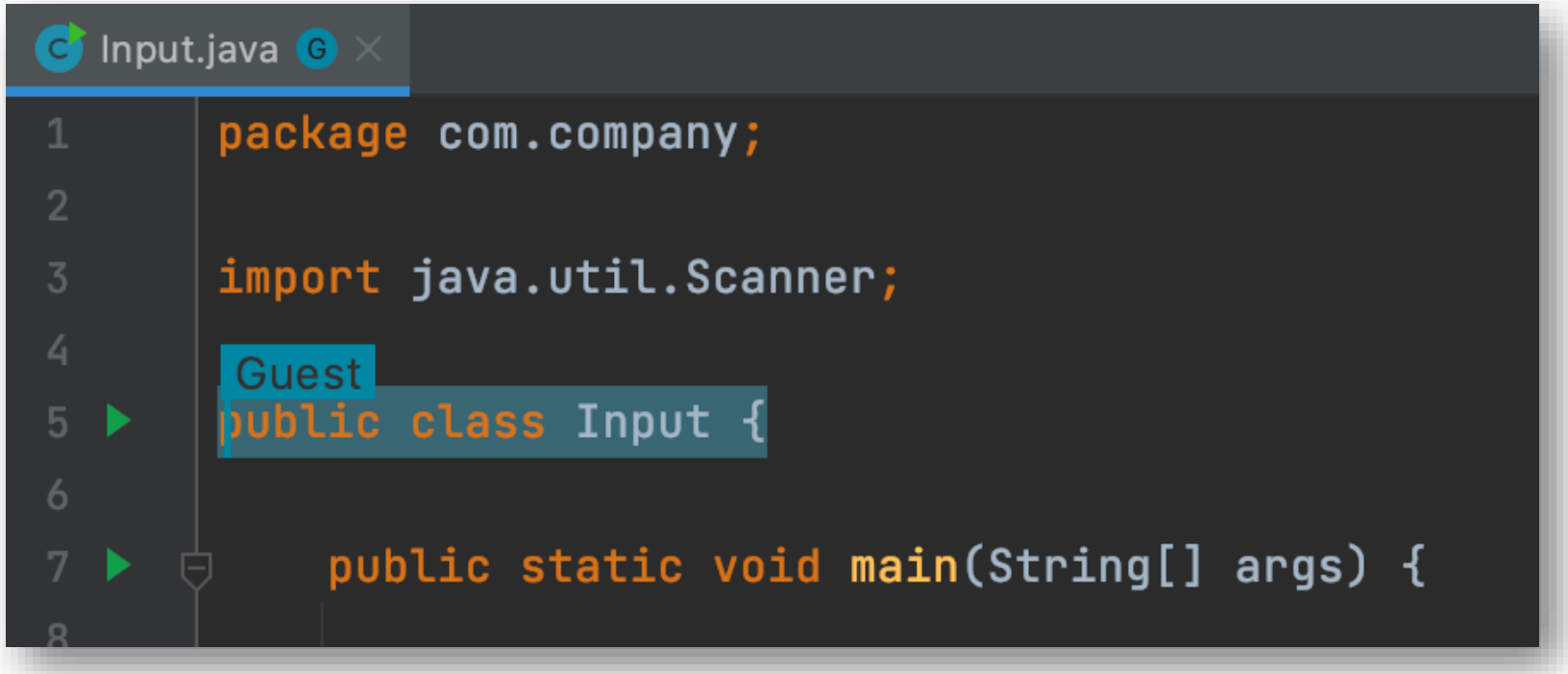
Search Actions & Create Shortcuts



Code With Me

Code With Me

- Code With Me is a PyCharm plugin that will allow you to code collaboratively, like in google docs.



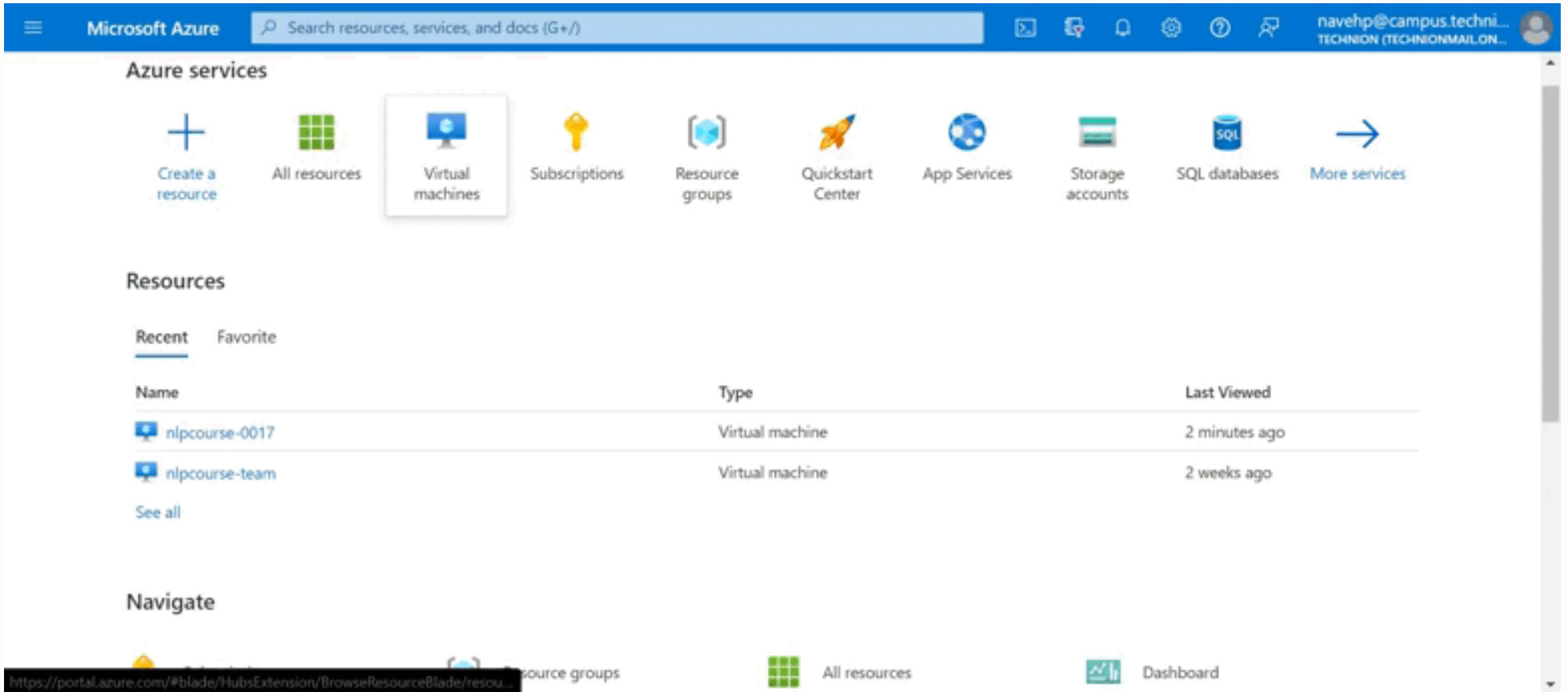
Working with VMs

Virtual Machines

- Pycharm Professional allows you to work on a remote virtual machine.
- The following slides will take you step by step in the process of configuring a remote machine in PyCharm.

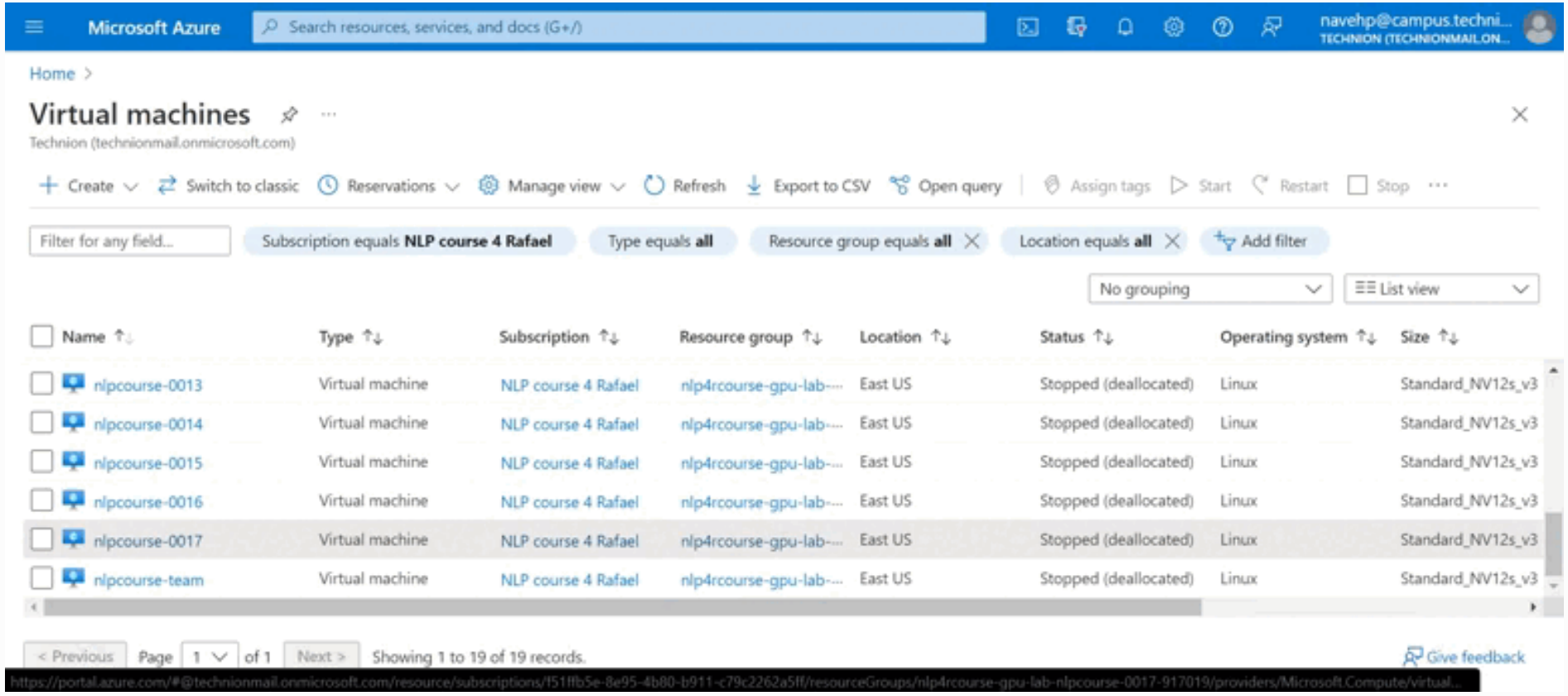
Turn on the Machine

- Visit the Azure Portal at https://portal.azure.com/#home
- Go into the **Virtual Machines** section.



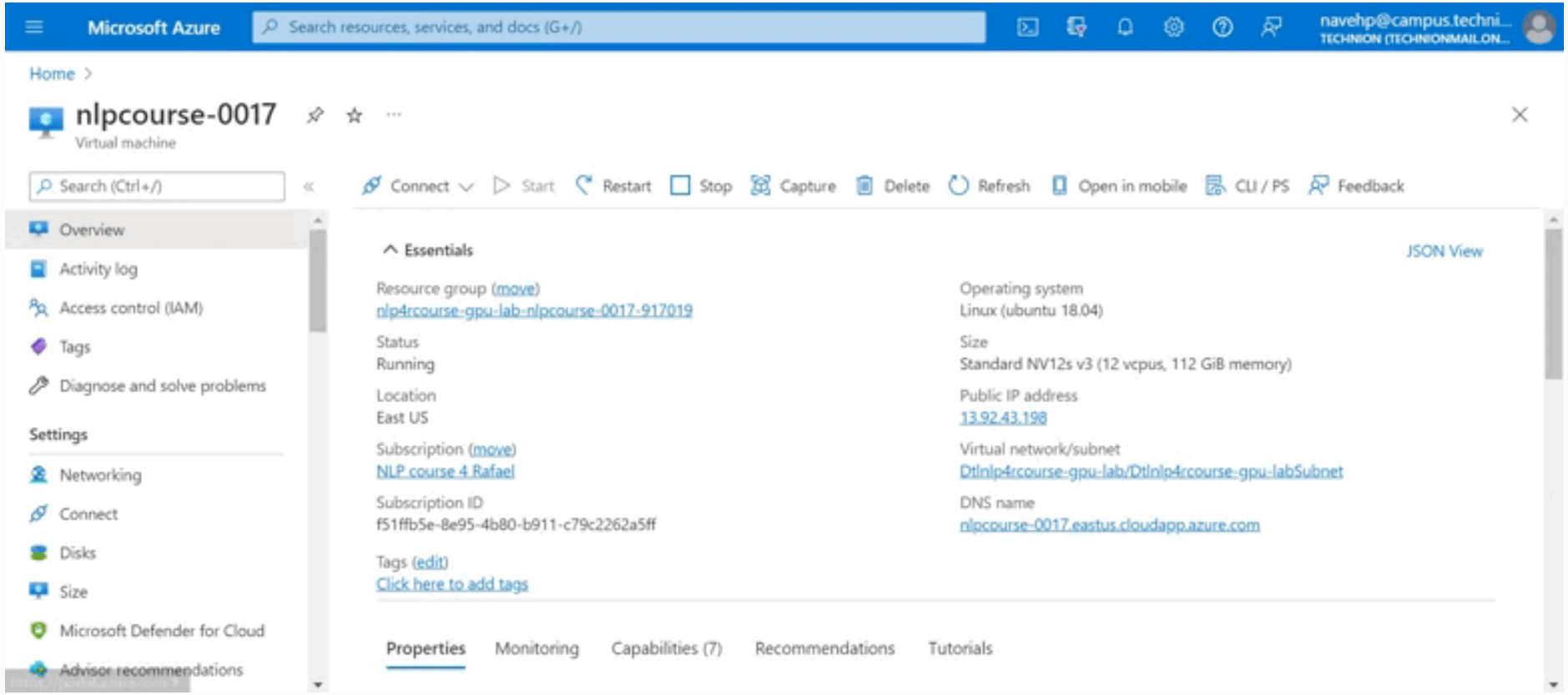
Turn on the Machine

- Check you're on the courses subscription.
- Enter your machine's page and click **start**.



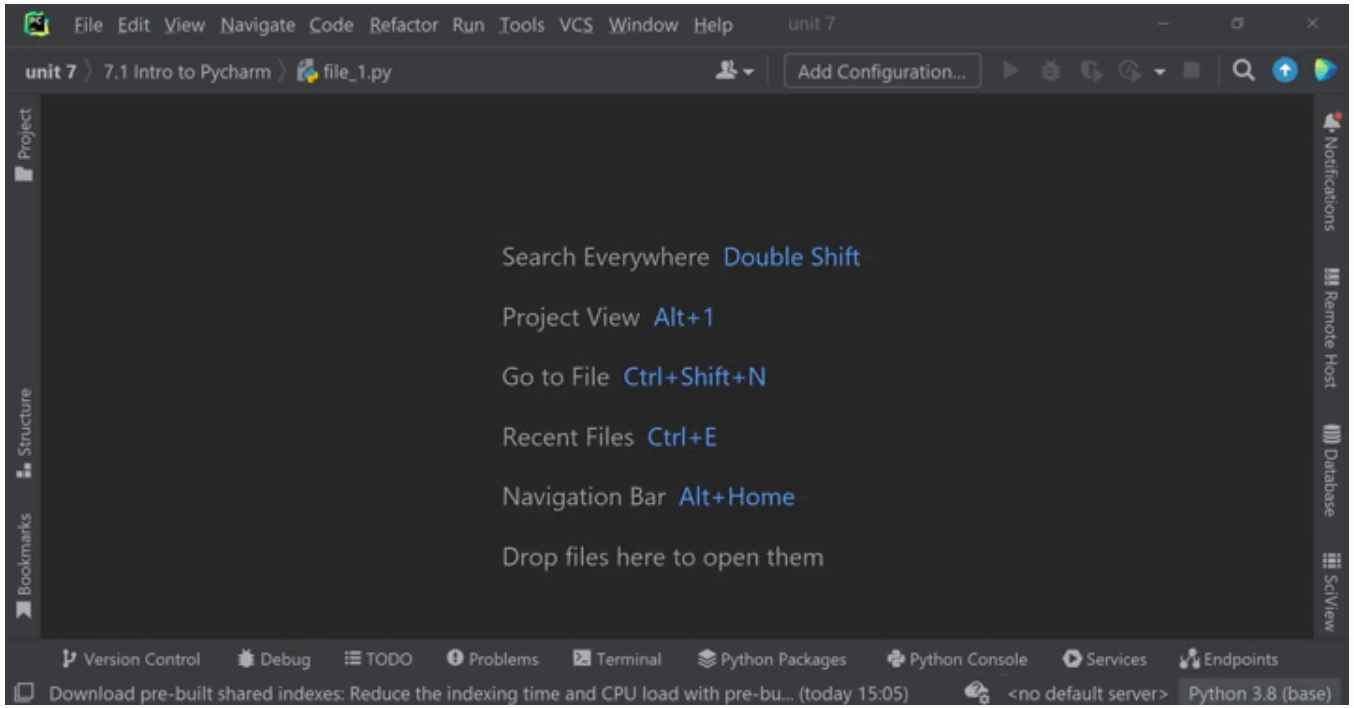
Turn on the Machine

- Once the machine has started, go into the **connect** section and copy the ssh connection command



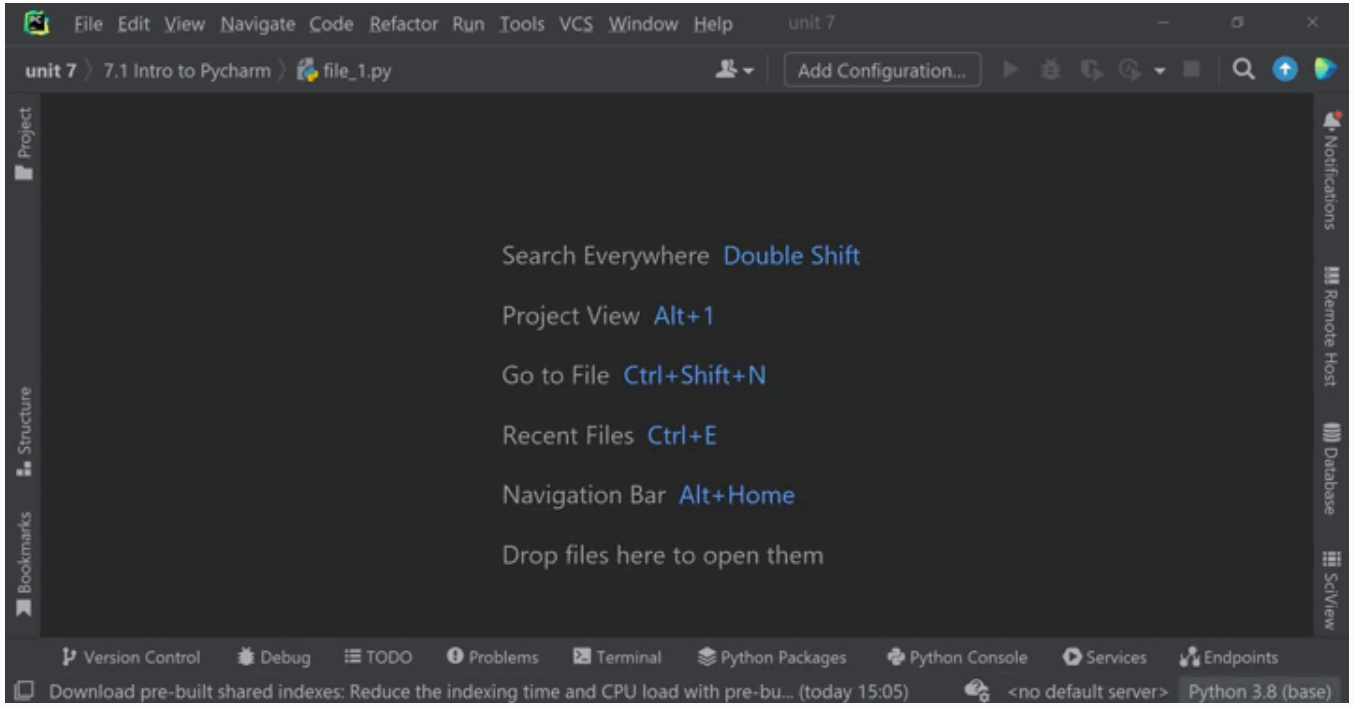
Configure The VM in PyCharm

- Configure a new ssh interpreter.
- The domain is part of the command you copied.
- The username should be **student**
- The password is *Technion2021!*



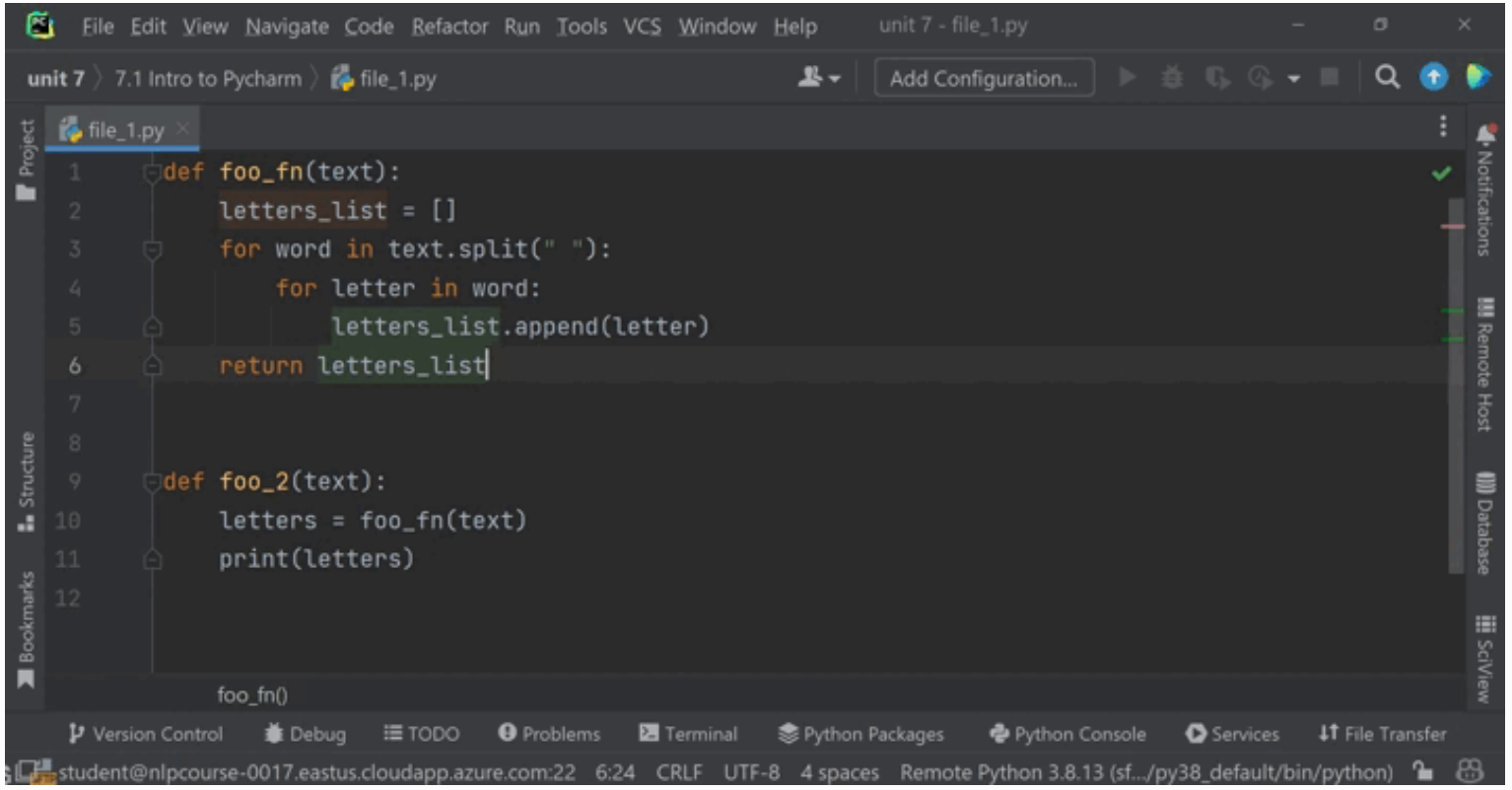
Configure The VM in PyCharm

- Once you connected to the machine, your interpreter will be found at /anaconda/envs/py38_default/bin/python
- In the **Sync Folders** section enter your local project directory and the remote one. An example of the remote path would be /home/student/project_dir



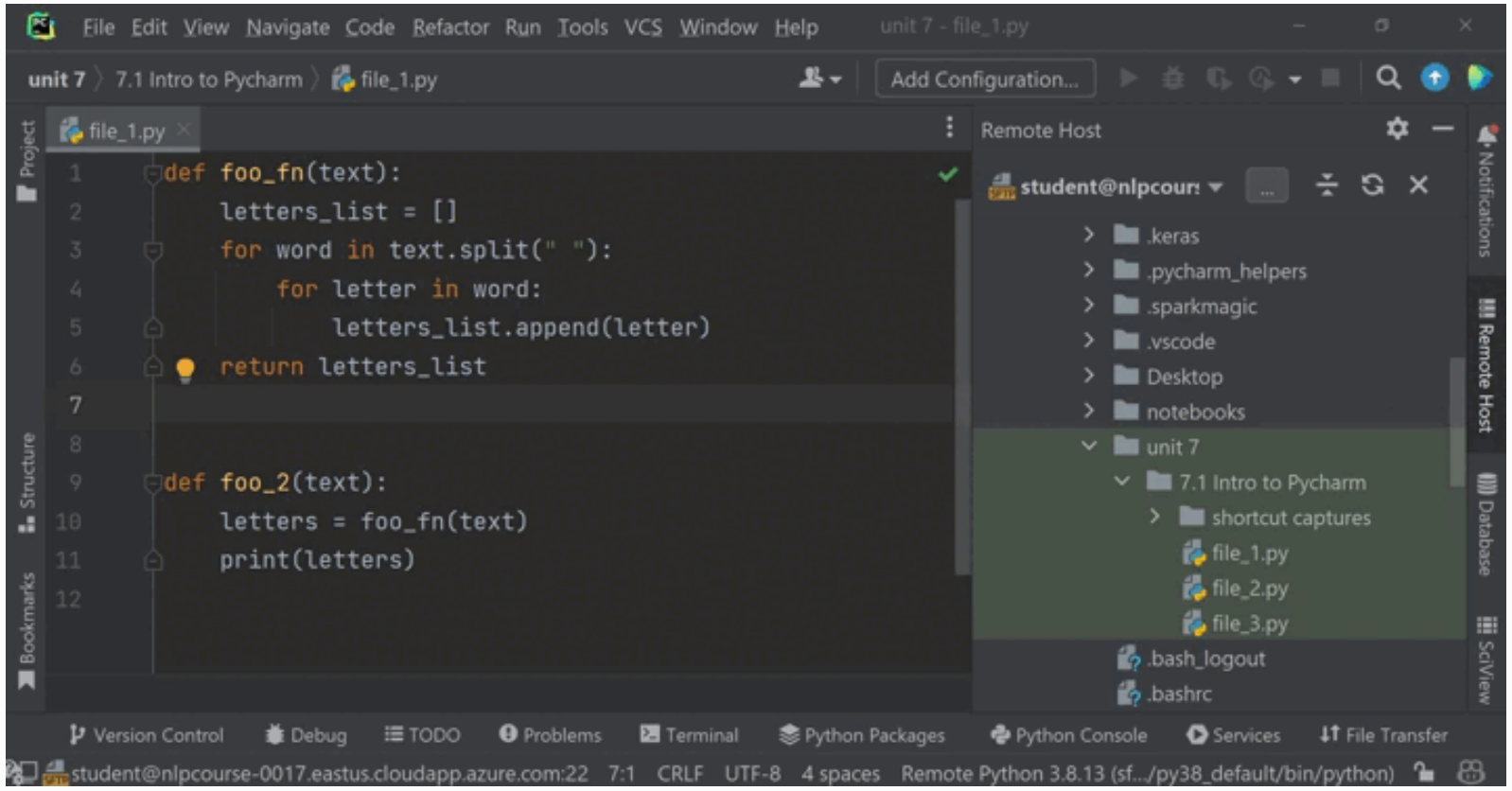
Uploading Files & Browsing

- Even though PyCharm will now automatically sync files from your local dir to the remote one, you can manually upload files using the Ctrl + Alt + Shift + x shortcut.
- You can browse the remote server via the **Remote Host** pane.



SSH Sessions

- You can now run an ssh session of your remote machine using tools > start ssh session.

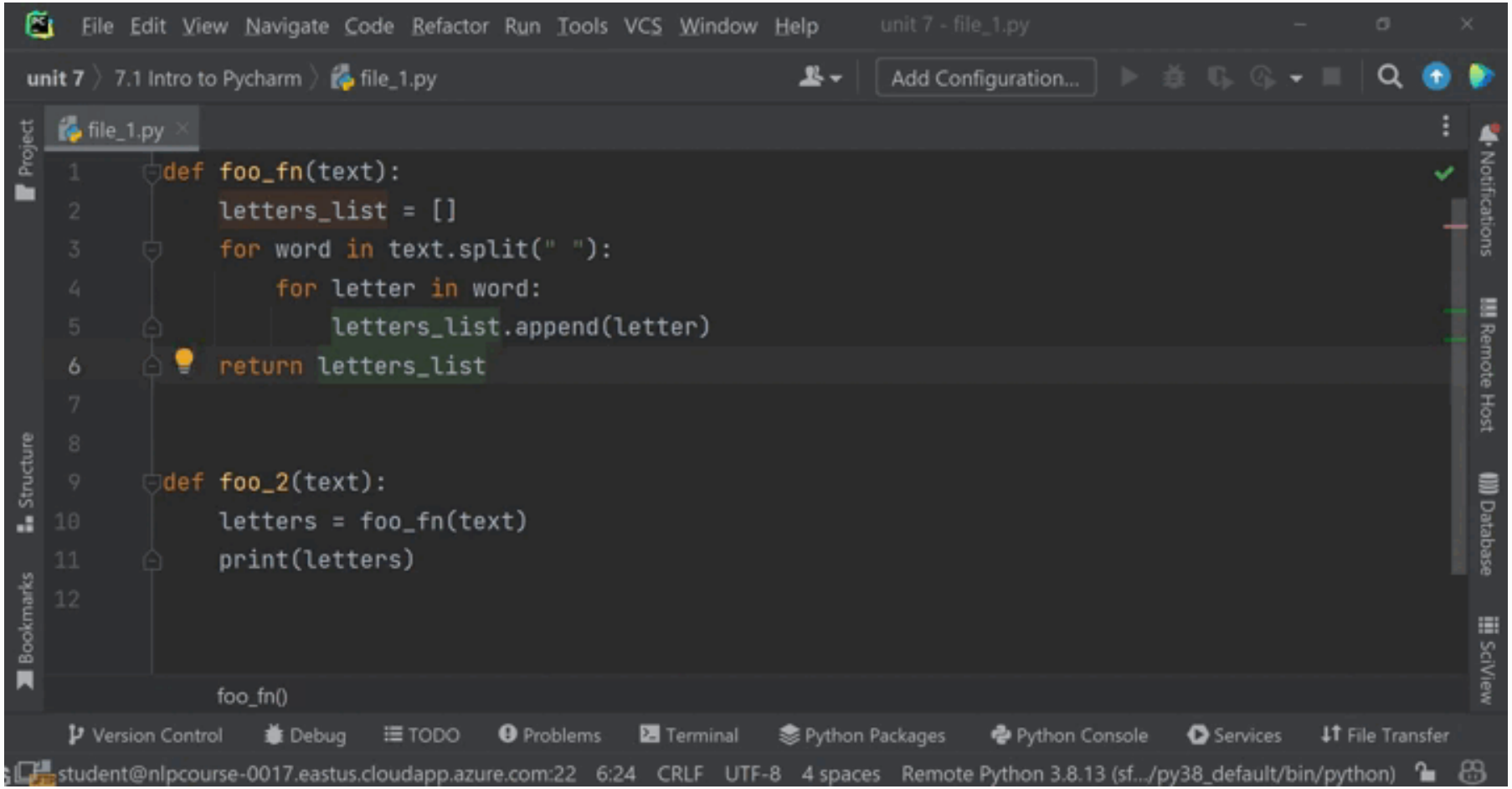


Question

- Configure you own machine!

Question

- Configure **Ctrl + Alt + Shift + s** as a shortcut for starting a ssh session.



Bash

Echo

```
student@nlpcourse-0017:~$ echo hi
hi
```

Variables

```
student@nlpcourse-0017:~$ msg="hi"
student@nlpcourse-0017:~$ echo "${msg}"
hi
student@nlpcourse-0017:~$ msg="bi"
student@nlpcourse-0017:~$ echo "${msg}"
bi
```

For Loops 1

```
student@nlpcourse-0017:~$ for msg in hi bi
> do
>     echo "${msg}"
> done
hi
bi
```

For Loops 2

```
student@nlpcourse-0017:~$ for i in {1..5}
> do
>   echo "$i"
> done
1
2
3
4
5
```

ls – listing directories

```
student@nlpcourse-0017:~$ ls
 Desktop   notebooks   'unit 7'
```

ls – listing directories

```
student@nlpcourse-0017:~$ ls 'unit 7'
'7.1 Intro to Pycharm'                  tutorial_04
'Lecture 4 - Intro to DL.pptx'         '~$Lecture 4 - Intro to DL.pptx'
'recitation_04 - intro to pytorch'
```

Mkdir – making directories

```
student@nlpcourse-0017:~$ mkdir tmp
student@nlpcourse-0017:~$ ls
 Desktop   notebooks   tmp  'unit 7'
```

cd – changing working directory

```
student@nlpcourse-0017:~$ cd tmp/
student@nlpcourse-0017:~/tmp$ ls
student@nlpcourse-0017:~/tmp$
```

Changing to parent directory

```
student@nlpcourse-0017:~/tmp$ cd ../
student@nlpcourse-0017:~$ ls
 Desktop   notebooks   tmp   'unit 7'
```

cp – copy files

```
student@nlpcourse-0017:~$ echo "hi" > tmp/hi.txt
student@nlpcourse-0017:~$ cp tmp/hi.txt tmp/bi.txt
student@nlpcourse-0017:~$ ls tmp/
bi.txt  hi.txt
```

cp – copy files

```
student@nlpcourse-0017:~$ cp tmp/ tmp2/
cp: -r not specified; omitting directory 'tmp/'
student@nlpcourse-0017:~$ cp -r tmp/ tmp2/
student@nlpcourse-0017:~$ ls tmp2/
bi.txt  hi.txt
```

mv – moving and renaming files

```
student@nlpcourse-0017:~$ ls
 Desktop   notebooks   tmp   tmp2  'unit 7'
student@nlpcourse-0017:~$ mv tmp2/ tmp3/
student@nlpcourse-0017:~$ ls
 Desktop   notebooks   tmp   tmp3  'unit 7'
```

rm – removing files

```
student@nlpcourse-0017:~$ ls tmp/
bi.txt  hi.txt
student@nlpcourse-0017:~$ rm tmp/bi.txt
student@nlpcourse-0017:~$ ls tmp/
hi.txt
```

rm – removing diretories

```
student@nlpcourse-0017:~$ ls
 Desktop   notebooks   tmp   tmp3  'unit 7'
student@nlpcourse-0017:~$ rm tmp3/
rm: cannot remove 'tmp3/': Is a directory
student@nlpcourse-0017:~$ rm -r tmp3/
student@nlpcourse-0017:~$ ls
 Desktop   notebooks   tmp  'unit 7'
```

pwd – getting absolute path

```
student@nlpcourse-0017:~$ pwd
/home/student
```

curl – get web pages

```
student@nlpcourse-0017:~$ curl https://gist.githubusercontent.com/simonsarris/9980a385af4f4c4d3967/raw/e4ba954591aaa5de8e4d36a4952e6229402e0fbd/poem.txt
 by Manuel Gutiérrez Nájera

I want to die as the day declines,
at high sea and facing the sky,
while agony seems like a dream
and my soul like a bird that can fly.
```

curl – save web pages

```
student@nlpcourse-0017:~$ curl https://gist.githubusercontent.com/simonsarris/9980a385af4f4c4d3967/raw/e4ba954591aaa5de8e4d36a4952e6229402e0fbd/poem.txt --output poem.txt
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   625  100   625    0     0  49856      0 --:--:-- --:--:-- --:--:-- 52083
student@nlpcourse-0017:~$ echo poem.txt
poem.txt
```

find – finding files

```
student@nlpcourse-0017:~$ find tmp/
tmp/
tmp/hi.txt
```

find – finding files

```
student@nlpcourse-0017:~$ find -name "hi.txt"
./tmp/hi.txt
```

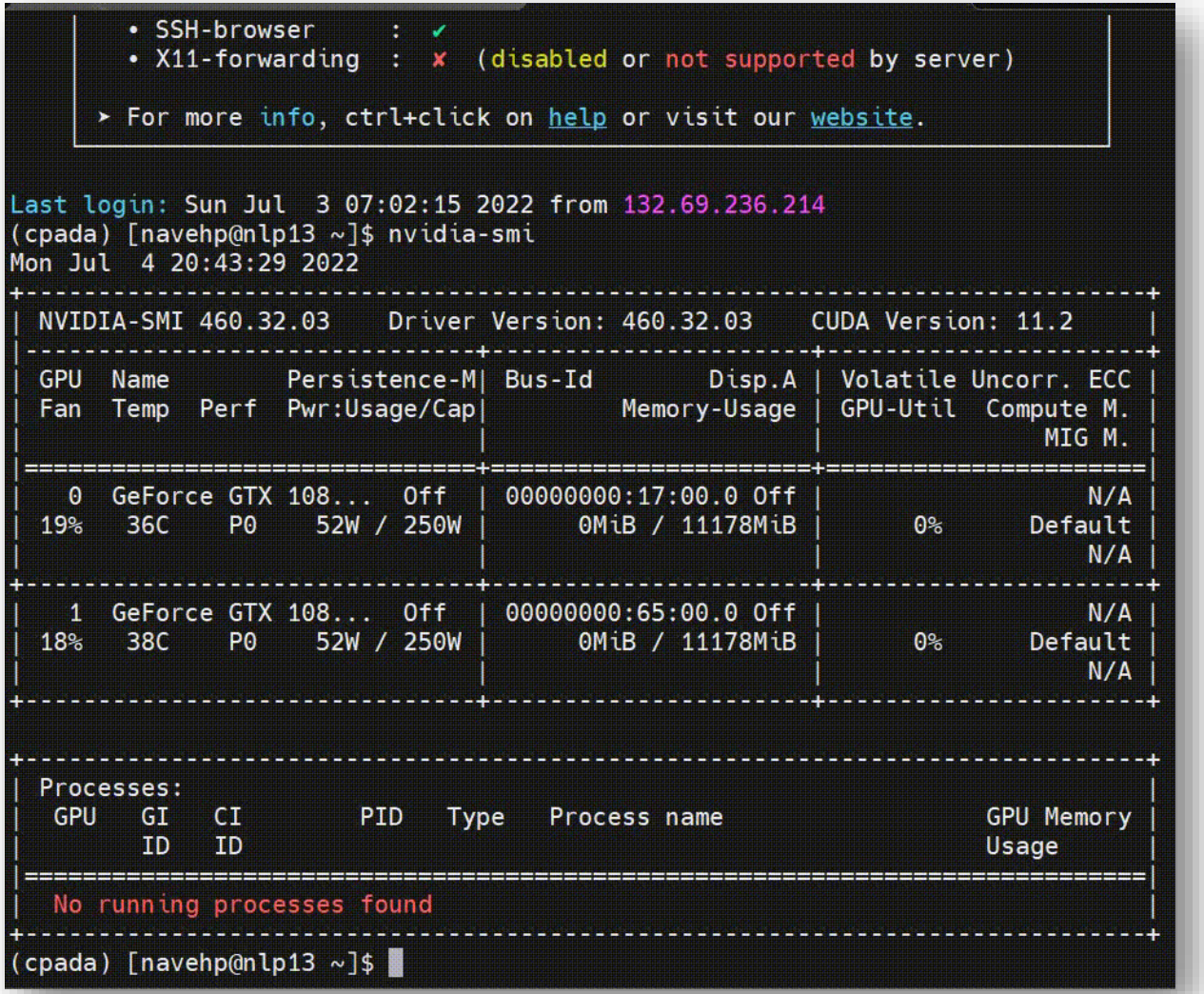
find – finding files

```
student@nlpcourse-0017:~$ find -name "*.txt"
./.pycharm_helpers/pydev/pydevd_attach_to_process/README.txt
./.pycharm_helpers/pydev/pydev_sitecustomize/__not_in_default_pythonpath.txt
./.pycharm_helpers/pydev/merge_pydev_pycharm.txt
./.pycharm_helpers/pydev/build_tools/pydevd_release_process.txt
./.pycharm_helpers/icon-robots.txt
./.pycharm_helpers/build.txt
./.pycharm_helpers/pycharm/teamcity/README.txt
./.pycharm_helpers/pycharm/django_manage_commands_provider/readme.txt
./.pycharm_helpers/pycharm/__jb.for_twisted/twisted/plugins/README.txt
./.pycharm_helpers/generator3/version.txt
./tmp/hi.txt
```

nvidia-smi – monitoring gpus

```
(cpada) [navehp@nlp13 ~]$ nvidia-smi
Mon Jul  4 20:43:29 2022
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.32.03    Driver Version: 460.32.03    CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  GeForce GTX 108...  Off  | 00000000:17:00.0 Off |                  N/A |
| 19%   36C    P0    52W / 250W |      0MiB / 11178MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  GeForce GTX 108...  Off  | 00000000:65:00.0 Off |                  N/A |
| 18%   38C    P0    52W / 250W |      0MiB / 11178MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

clear – clearing outputs



aliasing – creating your own commands

```
student@nlpcourse-0017:~$ alias nvid="nvidia-smi"
student@nlpcourse-0017:~$ nvid
Mon Jul  4 20:49:51 2022
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 470.103.01   Driver Version: 470.103.01   CUDA Version: 11.4     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
```

vi – file editing

- The vi command allows you to inspect and edit files inside bash.
- Run *vi file_path* to enter the vi editor.
- Vi has two modes: command and insert mode.
- To enter insert mode, press *i*.
- After editing your file, press *escape* to exit insert mode.
- You can quit your file in the following ways:
    - :q – quit an unchanged file.
    - :wq – write and quit a changed file.
    - :q! – force quitting a changed file without saving.

Screen – running processes in the background

- Many times, we would like to run processes in the background and have them not shut down once we disconnect ourself from the machine.
- Screen sessions are bash sessions that continue to run in the background.
- To create a new screen session run

  screen –S session_name
- To detach from a screen session press

  Ctrl a + d
- To reattach to a screen session run

  screen –r session_name

Screen – running processes in the background

- To kill a screen session run

screen –XS session_name quit

Or press **Ctrl a + k**

- Scrolling isn't available inside screen sessions.
- To move around, enter copy mode by pressing

Ctrl a + [

Sudo

- To install things in bash you need **sudo privileges**.
- You do not have sudo privileges on your machines, but if you run into something you need, contact us.

bash scripts

- You can place bash commands in a shell script and run it:

my_script.sh

```
1  echo "This is my script"
2  mkdir some_dir
3  echo "Made a directory called some_dir"
4  rm -r some_dir
5  echo "Removed the directory called some_dir"
```

```
student@nlpcourse-0017:~$ sh my_script.sh
This is my script
Made a directory called some_dir
Removed the directory called some_dir
```

Question

- Write a bash script called “download_alice.sh” that:
  - Creates a directory called *alice*.
  - Saves the alice book from github to the file *alice_book.txt* inside the dir.
  - Renames the file to *alice.txt*.
  - Prints the first 3 lines from the book (search the web!)
  - Prints informative messages between commands.
- Run the script

Solution

download_alice.sh

```
mkdir alice
echo "created the alice dir"
curl https://gist.githubusercontent.com/phillipj/4944029/raw/75ba2243dd5ec2875f629bf5
--output ./alice/alice_book.txt
echo "Downloaded alice_book.txt"
cp ./alice/alice_book.txt ./alice/alice.txt
echo "Renamed alice_book.txt to alice.txt"
echo "First 3 line of the book:"
head -3 ./alice/alice.txt
```

Solution

```
student@nlpcourse-0017:~$ sh download_alice.sh
created the alice dir
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  145k  100  145k    0     0  6339k      0 --:--:-- --:--:-- --:--:-- 6308k
Downloaded alice_book.txt
Renamed alice_book.txt to alice.txt
First 3 line of the book:
Alice's Adventures in Wonderland


                ALICE'S ADVENTURES IN WONDERLAND
```

Coding Practices

How to Write SOLID Code

- **S**ingle-responsibility principle (SRP)
- **O**pen–closed principle (OCP)
- **L**iskov substitution principle (LSP)
- **I**nterface Segregation Principle (ISP)
- **D**ependency Inversion Principle (DIP)

How to Write SOLID Code

- **Single-responsibility principle (SRP)**
- **Open–closed principle (OCP)**
- **Liskov substitution principle (LSP)**
- **I**nterface Segregation Principle (ISP)
- **D**ependency Inversion Principle (DIP)

Single-Responsibility Principle

Every component of your code should have one and only one responsibility

Open-Closed Principle

You should not need to modify the code you have already written to accommodate new functionality, but simply add what you now need.

Liskov Substitution Principle

Derived classes must be substitutable for their base classes

How to Write DRY Code

Don't **R**epeat **Y**ourself

Every piece of knowledge or logic must have a single, unambiguous representation within a system.

Example Project

Debugging

PyCharm Debugger

Live demonstration. A tutorial can be found here.

Some More Good Practices

- Splitting code into logical files.
- utils.py file.
- consts.py file.
- Using pathlib for paths.
- Documentation
- Configurable Code (e.g. YAML)
- Using argparse to run code from shell.
- Reformatting the code to PEP-8: Ctrl + Alt + L in PyCharm

Loggers

Wandb

- Live demonstration.
- A tutorial can be found here.
- **Sweeps** – We will use sweeps late on in the course. A tutorial can be found here.
- **Clear ML** is another (Israeli) logger that has some great features.

Tutorial

Task

- Take the existing project and add the following features:
  - Choice between an RNN/LSTM/GRU in the model.
  - Choice if to load pretrained embedding weights or not.
  - Whether to print the loss and accuracy in the evaluation loop.
  - Write evaluation results to csv
- Bonus: Create a hyperparameter search script that finds good hyperparameters.