Introduction to Python and Pytorch

TA: Lian Zhang

Email: luz244@psu.edu

1. Anaconda
2. Python
3. Pytorch
4. Jupyter Notebook
5. Use Jupyter Notebook to do programing with Python and Pytorch (reading material):

    Example 1: Plot the curve of a function

    Example 2: Root finding

    Example 3: Solve Ax=b

    Example 4: Calculate the derivative of a function

    Example 5: Find the maximum and minimum of a list of numbers

1. Anaconda

Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. Package versions are managed by the package management system conda. The Anaconda distribution includes data-science packages suitable for Windows, Linux, and MacOS.

```
Install Anaconda (Recommend: Python 3.7 version, Command Line ve
rsion) from
https://www.anaconda.com/distribution/
```

2. Python

Python is a high-level, general-purpose programming language. For examples:

(1) Create a calculator (calculate tips, annual tax, monthly expense...)

(2) Solve mathematical problems (root finding, plot some functions, solve Ax=b...)

(3) Say "hello" to Machine Learning

(4) Data mining (find all the online information which includes your name...)

(5) Create personal webpage

(6) Design games

...

Python is very powerful! If you want to do something with Python, just Google it!

Note that if you have already downloaded Anaconda, Python will be automaticly installed.

3. PyTorch

Pytorch is an open source machine learning library, used for applications such as computer vision and natural language processing. It is developed by Facebook's AI Research lab. It is free and open-source software.

```
Install Pytorch from https://pytorch.org/ , or use

conda install pytorch torchvision -c pytorch
```

4. Jupyter Notebook

Jupyter Notebook is a web-based interactive computational environment for creating Jupyter notebook documents. A Jupyter Notebook document contains an ordered list of input/output "cells" which can contain code, text (using Markdown), mathematics, plots and rich media, usually ending with the ".ipynb" extension.

```
A Jupyter Notebook can be converted to a number of open standard
```

4. Jupyter Notebook

Jupyter Notebook is a web-based interactive computational environment for creating Jupyter notebook documents. A Jupyter Notebook document contains an ordered list of input/output "cells" which can contain code, text (using Markdown), mathematics, plots and rich media, usually ending with the ".ipynb" extension.

```
A Jupyter Notebook can be converted to a number of open standard
output formats (HTML, presentation slides, LaTeX, PDF, ReStructu
redText, Markdown, Python) through "Download As" in the web inte
rface, via the nbconvert library or "jupyter nbconvert" command
line interface in a shell. (Student can do their hw with Jupyter
notebook, then submit a .ipynb file. Then the grader can run the
code)
```

e.g. This is an exapmle of Markdown cell with "Tex" equation.

$$f(x) = x^2$$

In [2]:

```
# e.g. This is an example of Python code cell.
2/8

# A line started with a pound sign means it is a comment line
```

Out[2]:

```
0.25
```

In [3]:

```
# How to use Pytorch
import torch

# We can initialize a matrix as a pytorch tensor:
A = torch.tensor([[1,2,3],[4,5,6],[7,8,9]])
print('A is', A)
```

```
A is tensor([[1, 2, 3],
        [4, 5, 6],
        [7, 8, 9]])
```

```
Install Jupyter Notebook from https://jupyter.org/install, or us
e
conda install -c conda-forge notebook

Run Jupyter Notebook:
Step 1: type "jupyter notebook" in command line
Step 2: Find the link shows in the command line
Step 3: Open the link with your browser
```

5. Use Jupyter Notebook to do programing with Python and Pytorch

Part A: Basic Python Language

Arithmetic

Like every programming language, Python is a good calculator. Run the block of code below to make sure the answer is right!

In [4]:

```
8 + 6*2*3 -1 + 2/3
```

Out[4]:

```
43.666666666666664
```

Question: How to calculate the power of a number?

In python, m ** n means m^n

In [5]:

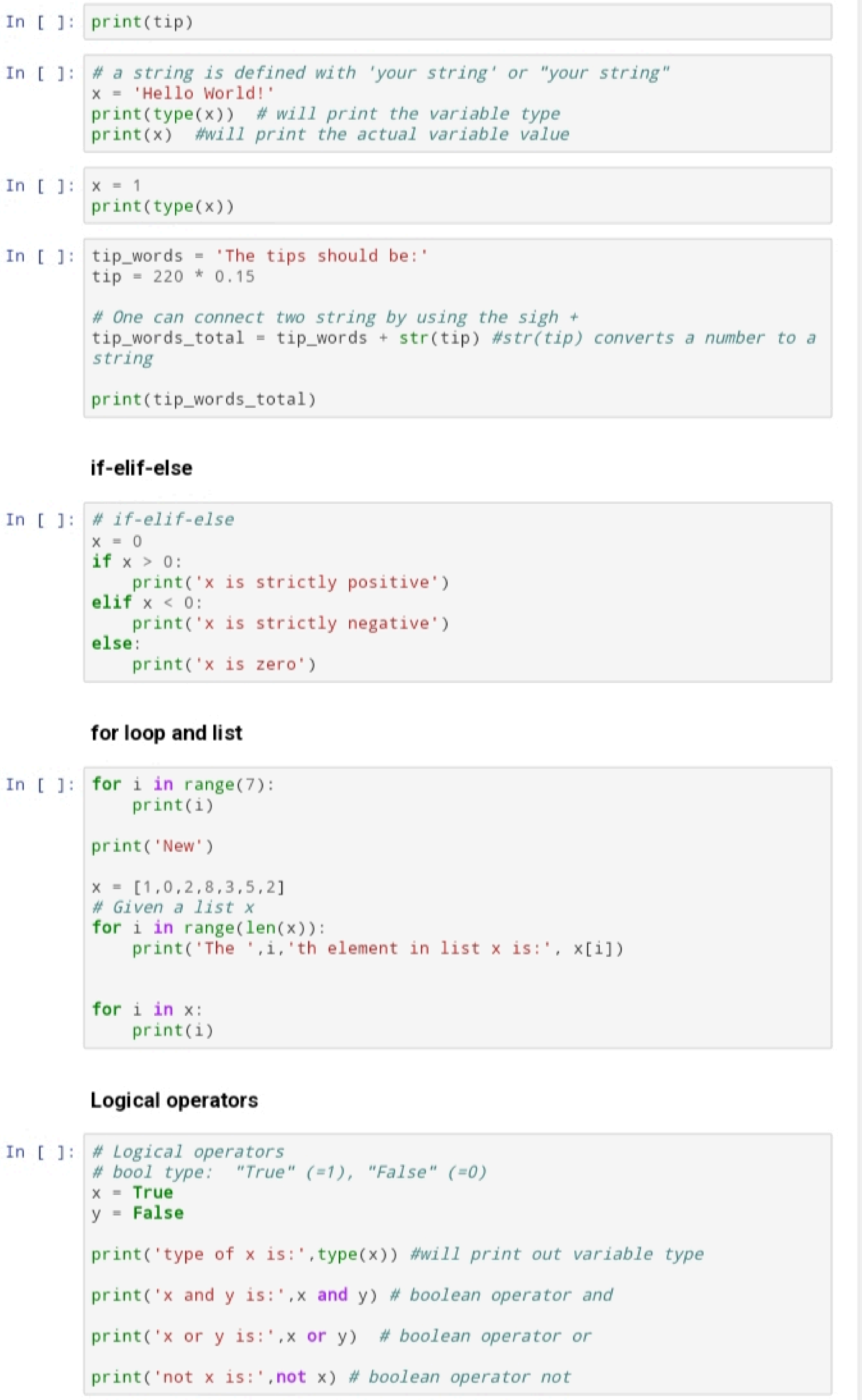
```
3 ** 2
2 ** 3
4 ** 2

if 4>2:
    print(4)
```

```
4
```

Pay attention: the rule of indent!

In [6]:

```
x=3
    y=3
```



In [ ]:
```python
print(tip)
```

In [ ]:
```python
# a string is defined with 'your string' or "your string"
x = 'Hello World!'
print(type(x))  # will print the variable type
print(x)  #will print the actual variable value
```

In [ ]:
```python
x = 1
print(type(x))
```

In [ ]:
```python
tip_words = 'The tips should be:'
tip = 220 * 0.15

# One can connect two string by using the sigh +
tip_words_total = tip_words + str(tip) #str(tip) converts a number to a string

print(tip_words_total)
```

if-elif-else

In [ ]:
```python
# if-elif-else
x = 0
if x > 0:
    print('x is strictly positive')
elif x < 0:
    print('x is strictly negative')
else:
    print('x is zero')
```

for loop and list

In [ ]:
```python
for i in range(7):
    print(i)

print('New')

x = [1,0,2,8,3,5,2]
# Given a list x
for i in range(len(x)):
    print('The ',i,'th element in list x is:', x[i])


for i in x:
    print(i)
```

Logical operators

In [ ]:
```python
# Logical operators
# bool type:  "True" (=1), "False" (=0)
x = True
y = False

print('type of x is:',type(x)) #will print out variable type

print('x and y is:',x and y) # boolean operator and

print('x or y is:',x or y)  # boolean operator or

print('not x is:',not x) # boolean operator not
```