Introduction to Python

OS Club - 05.04.2021

In this presentation…

- What is Python?
- Why use Python?
- Where is Python used?
- Getting Started with Python
- Hello world!
- if, for, while - Looping and branching
- Lists and dictionaries - Data structures
- Packages - pip
- File I/O - Dir listing

What is Python?

- General purpose programming language
- Interpreted
- Dynamically typed
  - i.e. No need to put “int” before variables
- Very high-level
- Multi-paradigm:
  - Procedural, object-oriented, functional
- One of the most popular languages

Why <3 Python?

- Very easy to learn!
  - Clean syntax, resulting in clean code
- Mature, with lots of external libraries
  - Probably already exists one that solves your problem
- Portable:
  - Can run on all major Operating Systems
    - Linux, Windows, MacOS
- Develop applications quicky
  - Rapid prototyping
  - Code is 2-10x shorter than C, Java (approximation)
- Zen of Python (import this)

Where is Python used?

- Web applications
- Machine Learning & Data Science
- Game development
- Automating tedious tasks
- Desktop Applications (GUI)
- Writing quick & dirty scripts
- Testing

Getting started with Python

- Python source files have .py extension
- I recommend that you use VS code to edit them
  - Syntax highlighting
  - Auto completion
- Once file is saved, run it like this in the terminal:
  - python3 <name_of_your_file>.py

Hello world!

```
# This is a comment
# Hello world example program!
print("hello world!")
```

Getting input from the user

```
# ask the user for their name
name = input("What is your name?")
print("hello", name)
```

- Note that there is no type declared before name
- but name has type ‘str’ (string)
- we can verify this via the type function

Formatting strings - f-strings

- Introduced in Python 3.6
- Elegant way to format strings (printf-like)
- String should start with f

```
name = 'david'
age = 100

formatted_string = f"My name is {name}, my age is {age}"
print(formatted_string)
```

Transforming a str to an int

```
age_str = input("What is our age?")
age = int(age_str)
print("your age is:", age)
```

- If an invalid number is given ValueError exception is raised and the program is terminated

ifs and elses

```
name = input("What is your name?")
print("hello", name)
if name == "acs david":
    print("its me")
else:
    print("its somebody else")
```

- Note colon (:) after if and else
- else can be missing
- Indentation in python is necessary

Boolean expressions

- Note: conditions inside if are evaluated based on Truthy / Falsy logic. E.g. Empty string is False, non-empty string is considered True
- Can negate conditions with not keyword, like !
- and keyword is equivalent of &&
- or keyword is equivalent of ||

for

```
# example with for.
for i in range(10):
    print("current i is:", i)
```

- range generates number from 0 until the number specified between the parentheses.

lists

```
# create an empty list
my_list = []

# add to list
my_list.append("my_string")

print(my_list)

my_list.append("another string")

# access first element
print(my_list[0])

# iterate over list
for s in my_list:
    print(s)
```

Exercise time!

Create a Python program that:

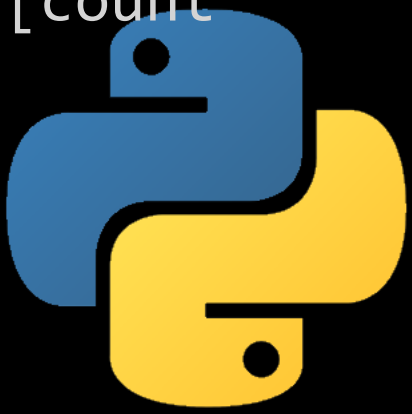
- Asks user for 4 integers, put them into a list
  - For each input, transform them to int
- Check if the number is even or not
  - You can use the % (modulo) operator
- And print only the even numbers from the list

Dictionaries - key value store

```
# create an empty dictionary
country_capitals = {}

# add elements to dictionary
country_capitals["Romania"] = "Bucharest"
country_capitals["UK"] = "London"

for country in country_capitals:
    print("Capital of", country, "is", country_capitals[count
ry])
```



functions

```
# define new functions with "def", followed by param definitions
def welcome_user(name, age):
    print("welcome", name, "you have", age, "years!")

# call functions like this.
welcome_user("David", 42)

def return_something():
    return 42

something = return_something()
print(something)
```

Exercise time!

Create a function that returns True if a number is prime and False otherwise

Using this function, create a program that:

- Asks the user to input a number (n)
- Create a dictionary that has keys numbers from 0 to (n), where value is “prime” or “not prime”
- print the contents of the dictionary.

Packages & pip

- Python has lots of useful preinstalled packages
- Community developed even more
- They can be downloaded with pip
  - Download pip, if you don’t have it already:
    - $ sudo apt install python3-pip
  - $ pip3 install <package_name>

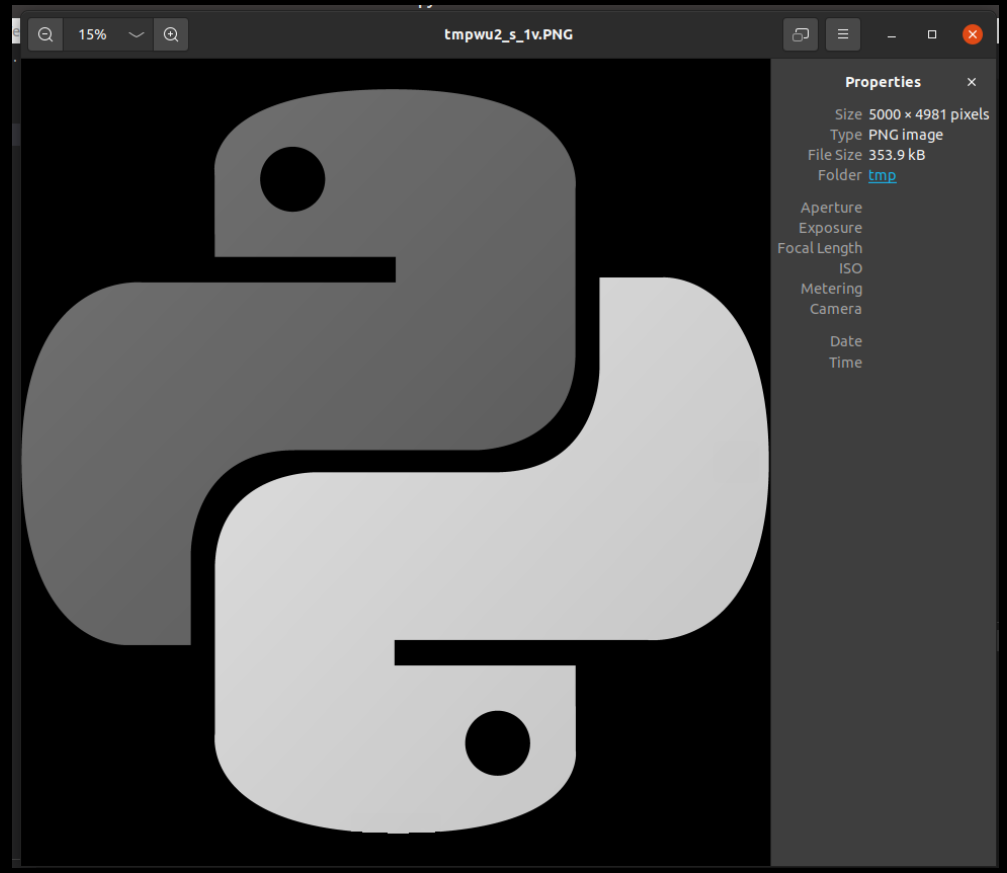
How to use packages

```
from PIL import Image

logo = Image.open("logo.png")

# convert to grayscale (L)
grayscale = logo.convert('L')

grayscale.show()
```



File I/O - reading

- Function names and their usage similar to OS's C API

```
file = open("test_file.txt", 'r')

# read all text in the file
content = file.read()
print(content)

# or alternatively read all lines. Returns list of lines
for line in file.readlines():
    print(line)

# we don't need the file anymore, clean up
file.close()
```

File I/O - writing

```
# open for writing
file = open("example_file.txt", 'w')

file.write("hello world!")

file.close()

# use context manager to automatically close open file:
with open("context_mgr.txt", 'w') as f:
    f.write("write from context manager!")
```

Directory listing - Non-rec

```
import os

current_directory_contents = os.listdir(".")
for content in current_directory_contents:
    print(content)
```

Directory listing - Recursive

```
import os

for root, dirs, files in os.walk("."):
    print("Directories in", root)
    for d in dirs:
        dir_path = os.path.join(root, d)
        print(dir_path)

    print("Files in", root)
    for f in files:
        file_path = os.path.join(root, f)
        print(file_path)
```

Exercise time!

Create a Python script that:

- Tries to find recursively files in a directory
- The file must contain two lines that start with the string “TEST”
- If such a file is found it’s path must be printed

Challenge (optional)

Implement requirements for assignment1 in Python

Tips and tricks:

- Access the list of arguments via sys.argv
- For list use os.listdir and os.walk
- For parse use open('path', 'rb') - read binary
- Use int.from_bytes to transform bytes to int
- Use file.seek(offset) if you need to move around the file, like with lseek

Bibliography

- https://docs.python.org/3/tutorial/
- https://www.tutorialspoint.com/python3/index.htm
- https://www.python.org/

Thank you for your Attention!

Any questions?