bianca.phelan@tudublin.ie

O Object
O Oriented
P Programming

S1 = Python with Bianca

S2 = Java with Bryan

DT228(TU856)/DT282(TU858) - 2

COMPUTER SCIENCE

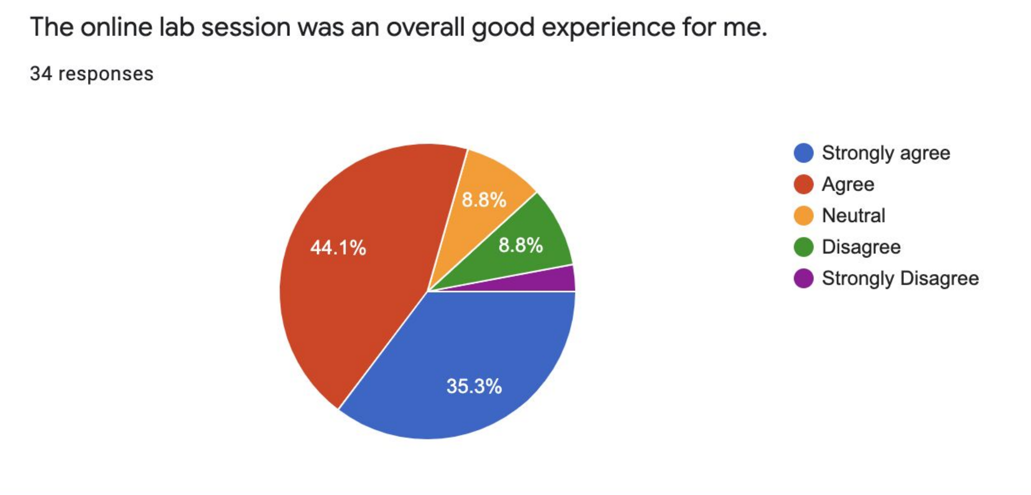
Week 2
Lab Revision and Python Foundations

Objectives

- Revise some concepts from the lab and general programming understanding
- Finish up on Wednesday’s tutorial

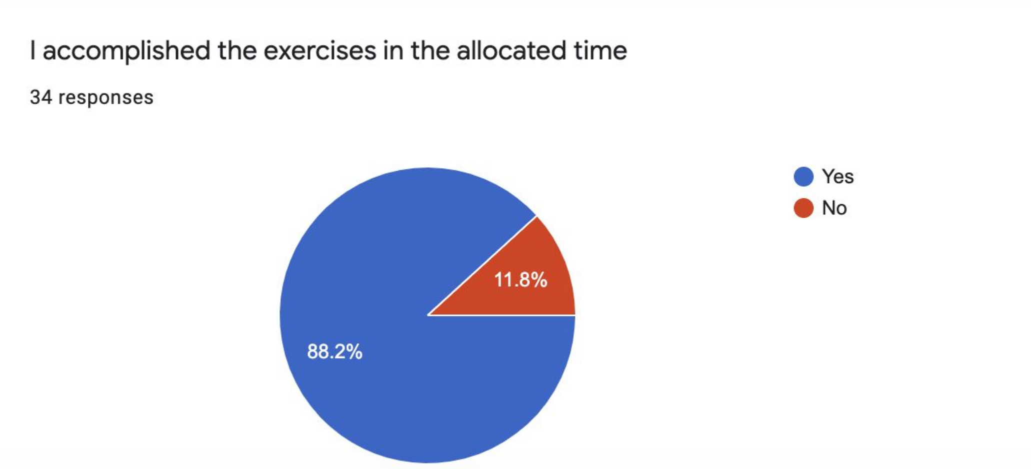
Lab Feedback

- Stay in the lab group with one tutor
- More windows support



The online lab session was an overall good experience for me.

34 responses

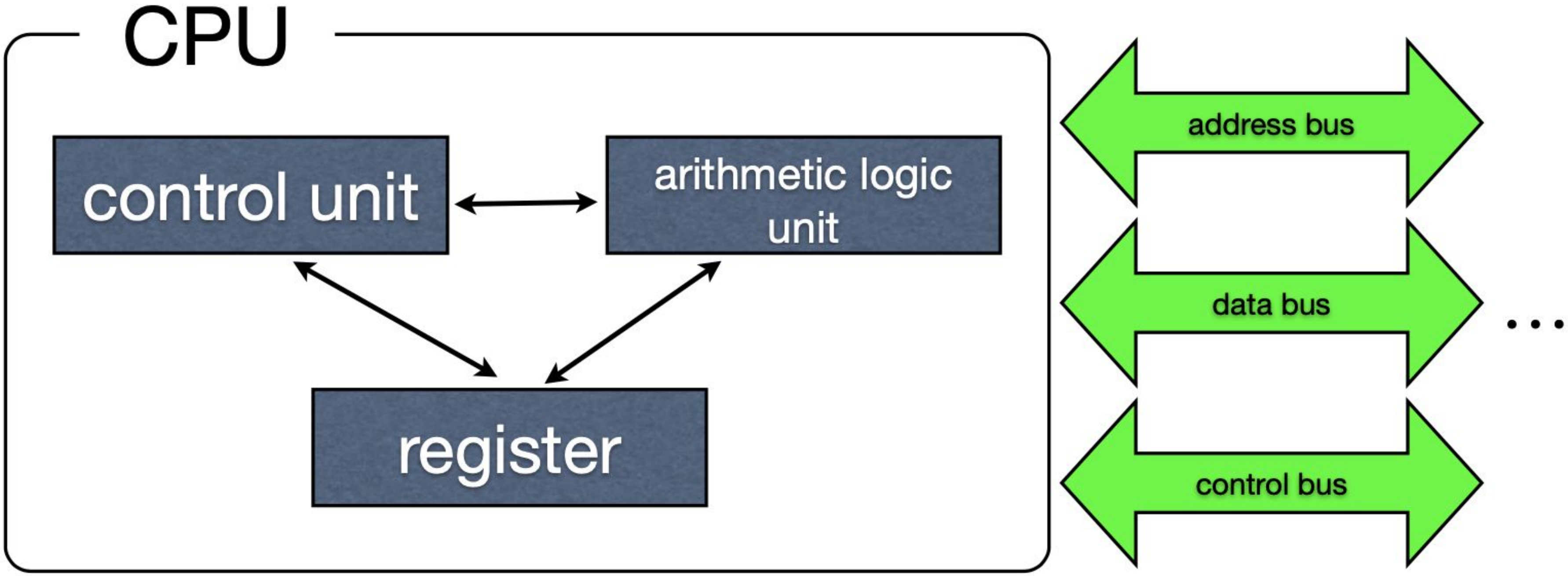


I accomplished the exercises in the allocated time

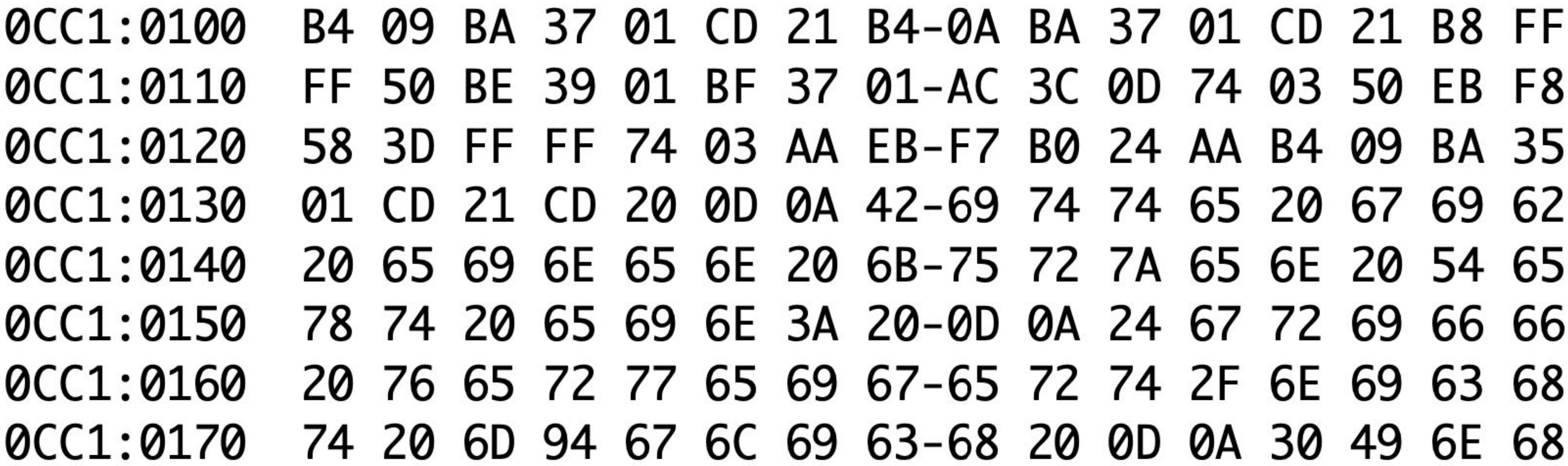
34 responses

Programming Background

Compilers, Interpreters and IDEs...

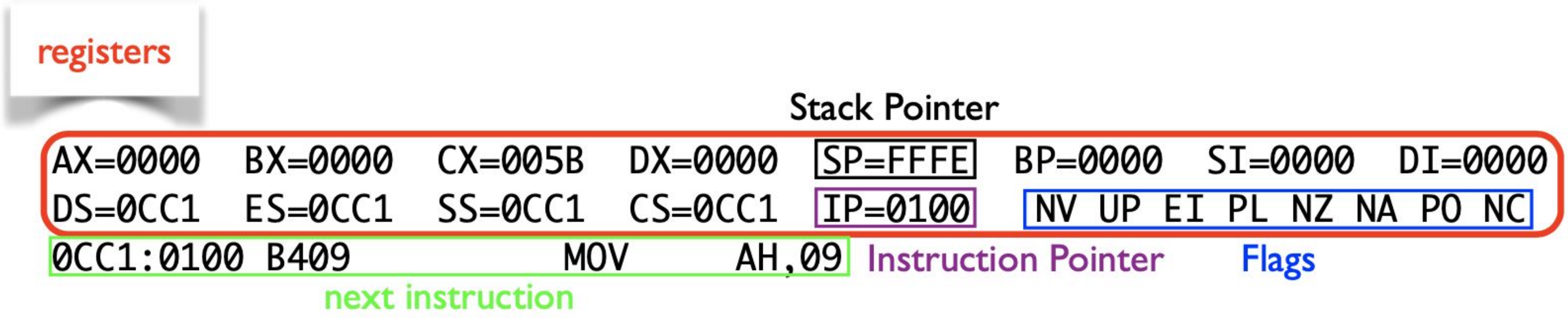


What the CPU Sees



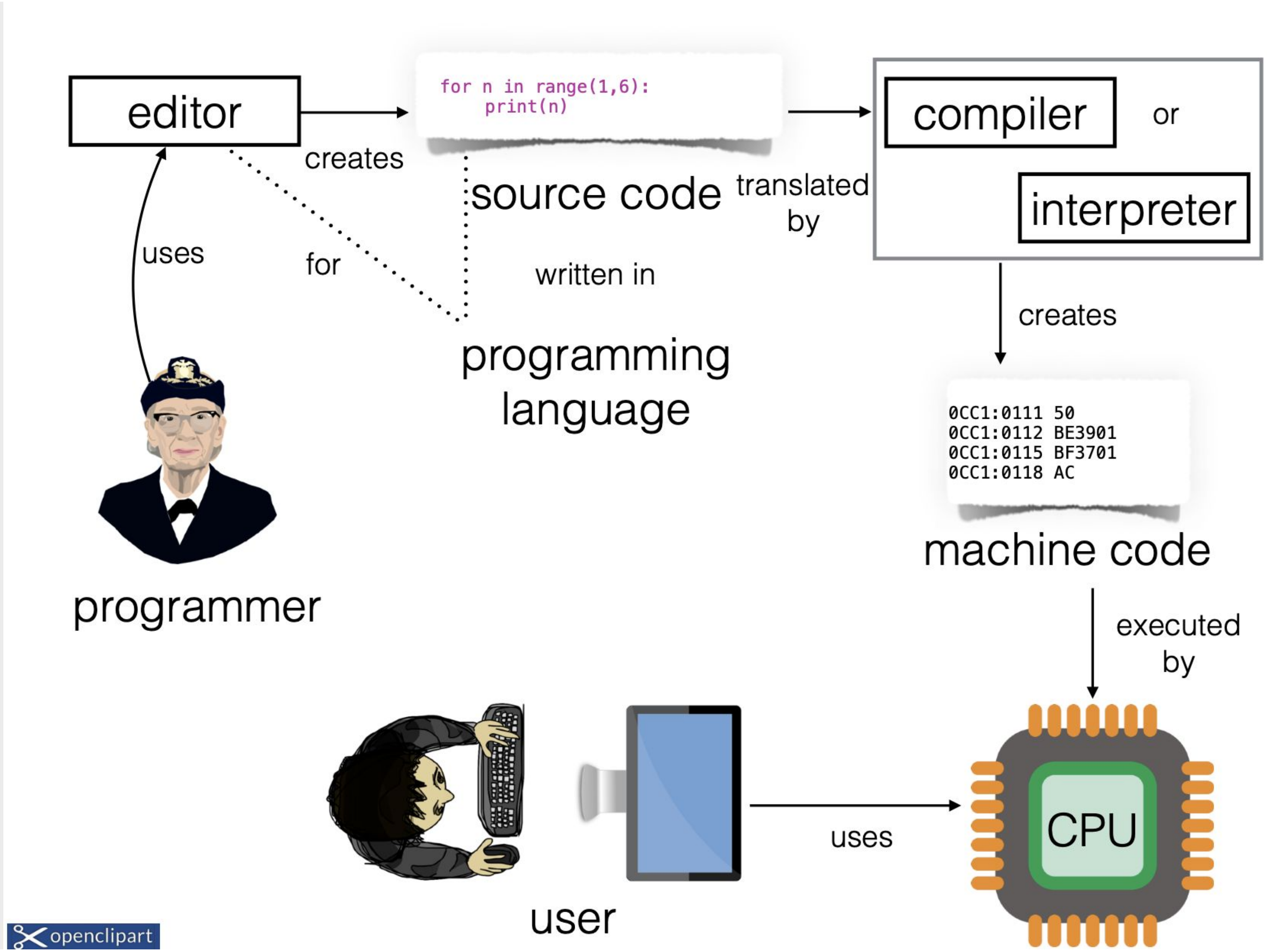
```
0CC1:0100  B4 09 BA 37 01 CD 21 B4-0A BA 37 01 CD 21 B8 FF
0CC1:0110  FF 50 BE 39 01 BF 37 01-AC 3C 0D 74 03 50 EB F8
0CC1:0120  58 3D FF FF 74 03 AA EB-F7 B0 24 AA B4 09 BA 35
0CC1:0130  01 CD 21 CD 20 0D 0A 42-69 74 74 65 20 67 69 62
0CC1:0140  20 65 69 6E 65 6E 20 6B-75 72 7A 65 6E 20 54 65
0CC1:0150  78 74 20 65 69 6E 3A 20-0D 0A 24 67 72 69 66 66
0CC1:0160  20 76 65 72 77 65 69 67-65 72 74 2F 6E 69 63 68
0CC1:0170  74 20 6D 94 67 6C 69 63-68 20 0D 0A 30 49 6E 68
```

What Happens Inside the CPU



```
AX=0000  BX=0000  CX=005B  DX=0000  SP=FFFE  BP=0000  SI=0000  DI=0000
DS=0CC1  ES=0CC1  SS=0CC1  CS=0CC1  IP=0100   NV UP EI PL NZ NA PO NC
0CC1:0100 B409          MOV     AH,09
```

How It Fits Together



editor

```
for n in range(1,6):
    print(n)
```

creates

source code

translated by

compiler

or

interpreter

uses

for

written in

programming language

creates

```
0CC1:0111 50
0CC1:0112 BE3901
0CC1:0115 BF3701
0CC1:0118 AC
```

machine code

programmer

executed by

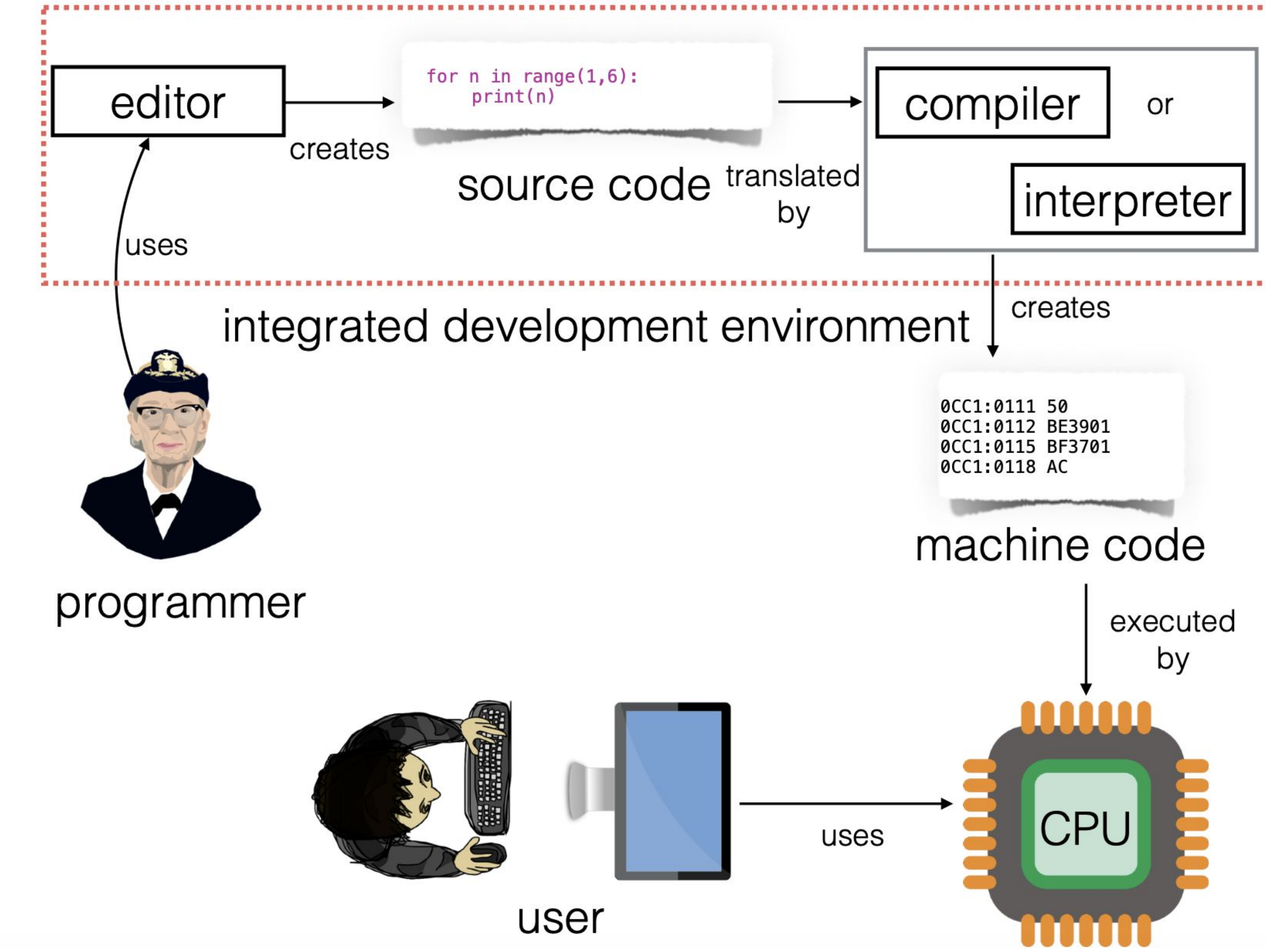
uses

CPU

user

openclipart

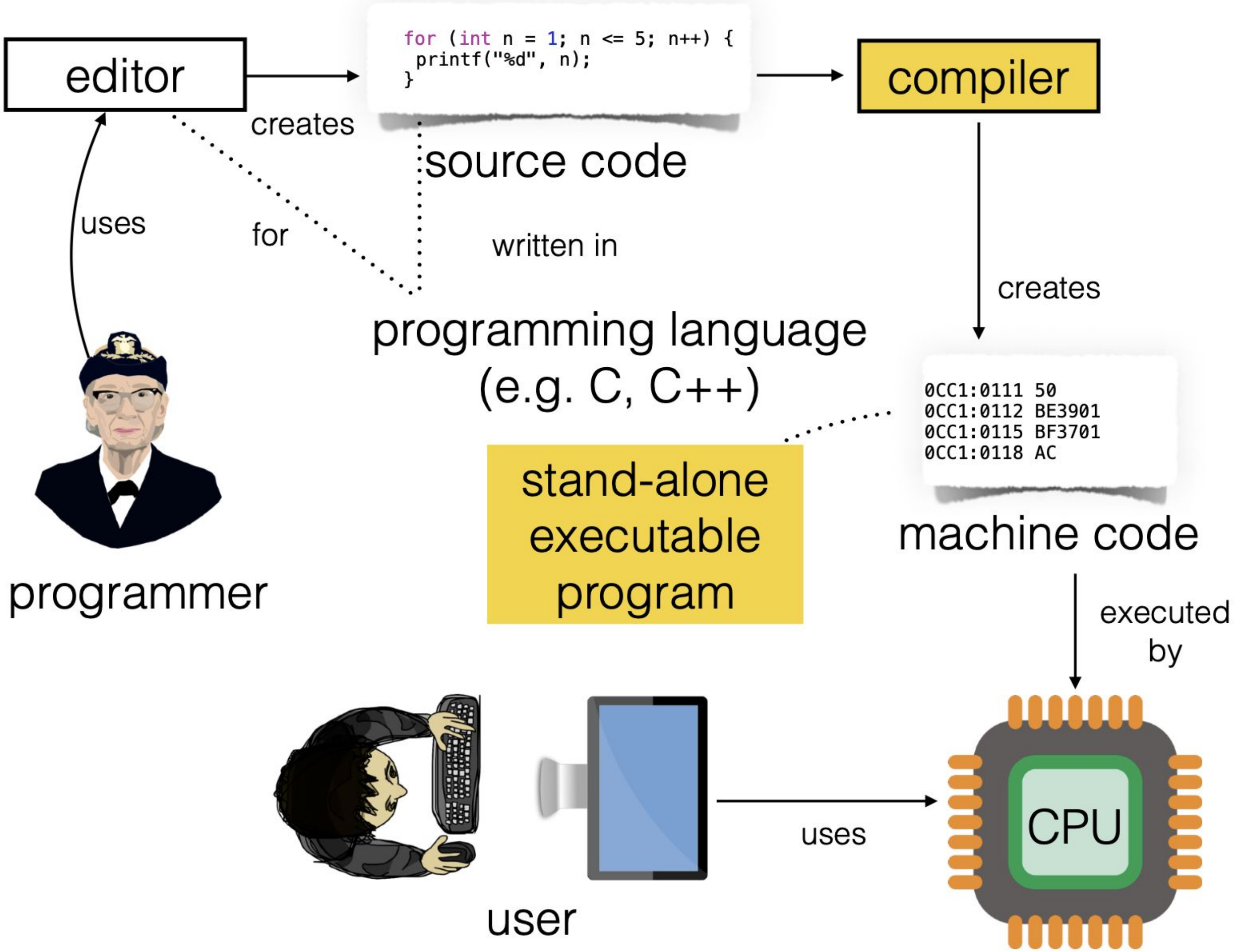
Integrated Development Environment



```
for n in range(1,6):
    print(n)
```

```
0CC1:0111 50
0CC1:0112 BE3901
0CC1:0115 BF3701
0CC1:0118 AC
```

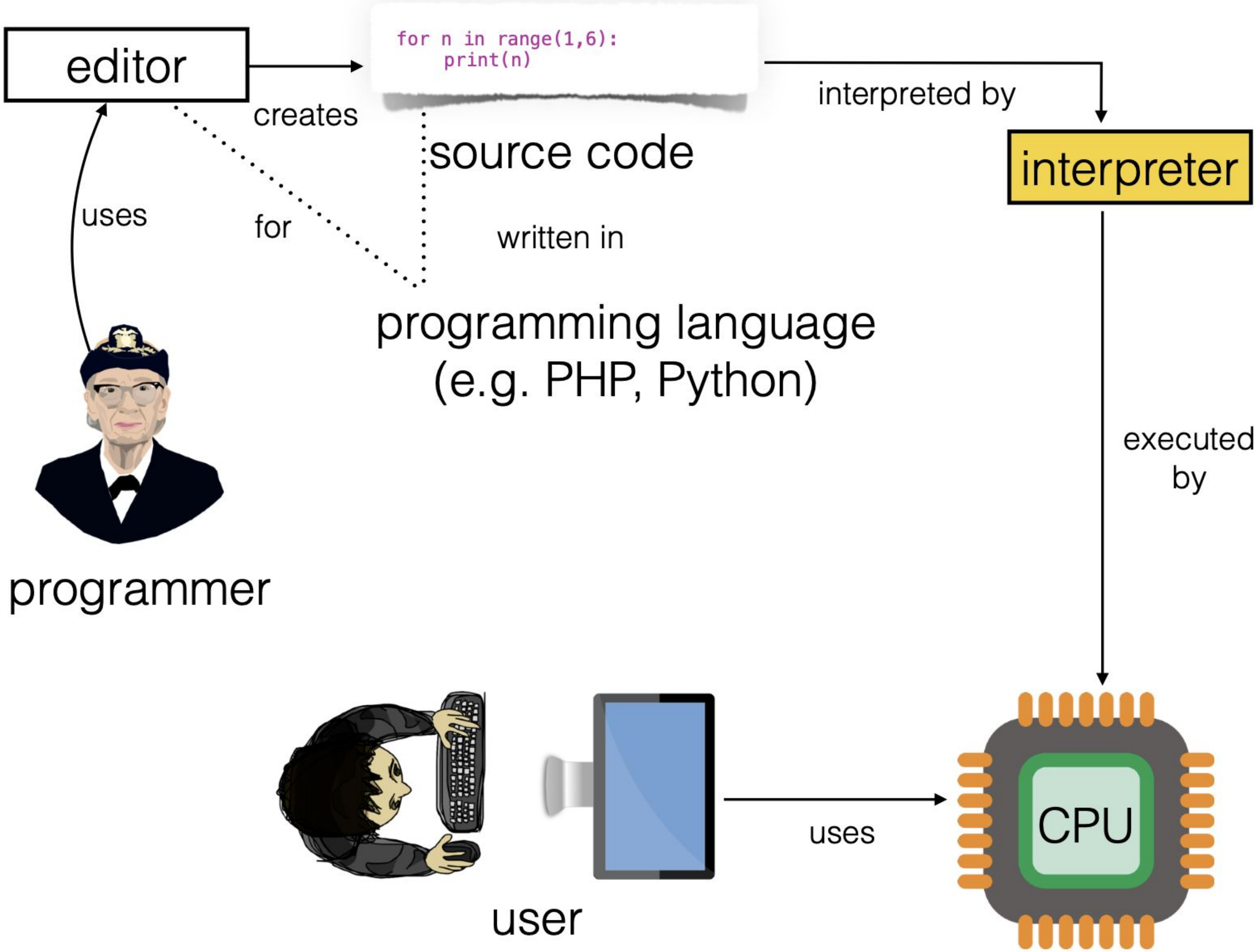
Compiler



```
for (int n = 1; n <= 5; n++) {
 printf("%d", n);
}
```

```
0CC1:0111 50
0CC1:0112 BE3901
0CC1:0115 BF3701
0CC1:0118 AC
```

Interpreter



editor

creates

```
for n in range(1,6):
    print(n)
```

source code

interpreted by

interpreter

uses

for

written in

programming language
(e.g. PHP, Python)

executed by

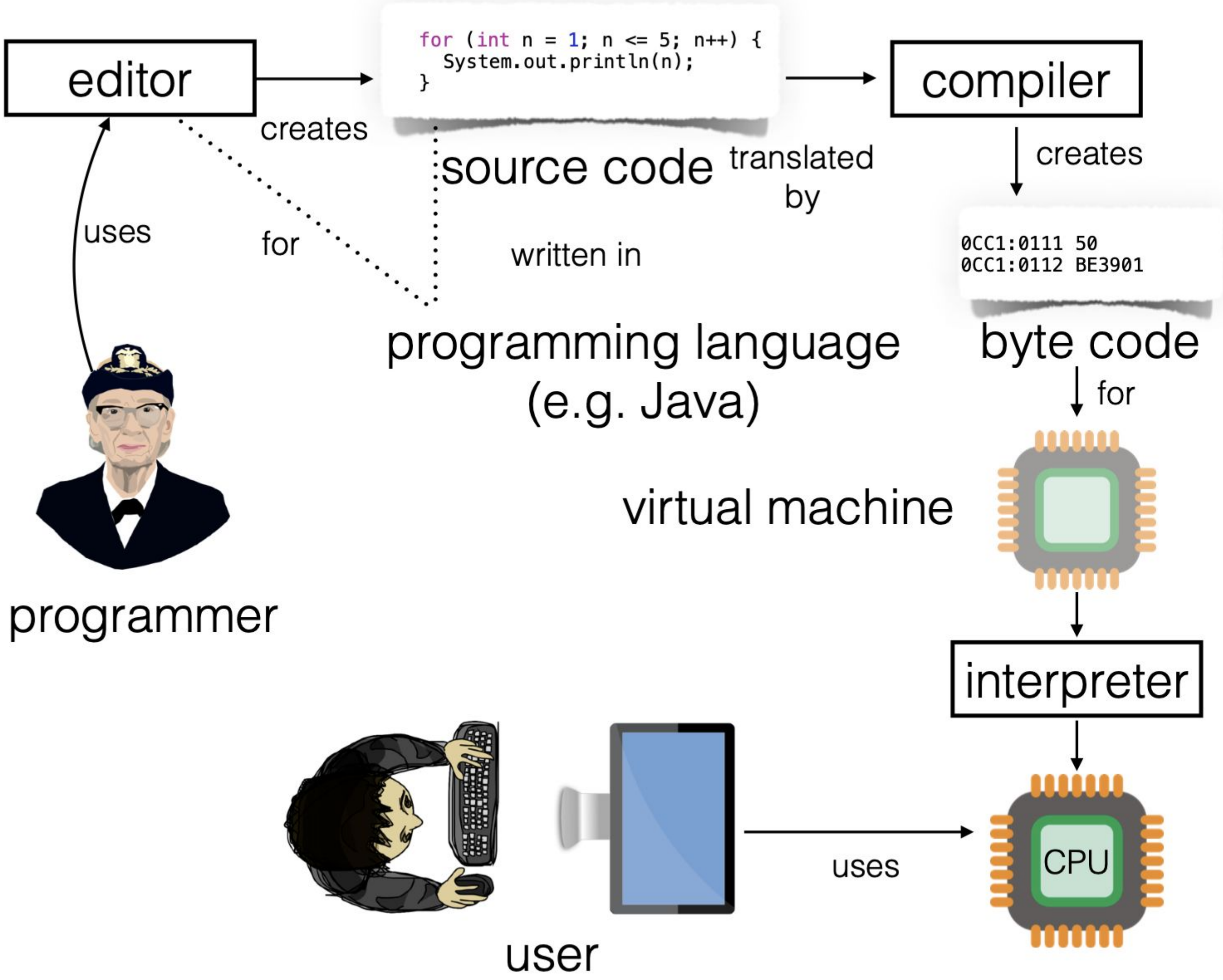
programmer

user

uses

CPU

Java Example



editor

creates

```
for (int n = 1; n <= 5; n++) {
  System.out.println(n);
}
```

source code

translated by

compiler

creates

```
0CC1:0111 50
0CC1:0112 BE3901
```

byte code

for

virtual machine

interpreter

uses

for

written in

programming language (e.g. Java)

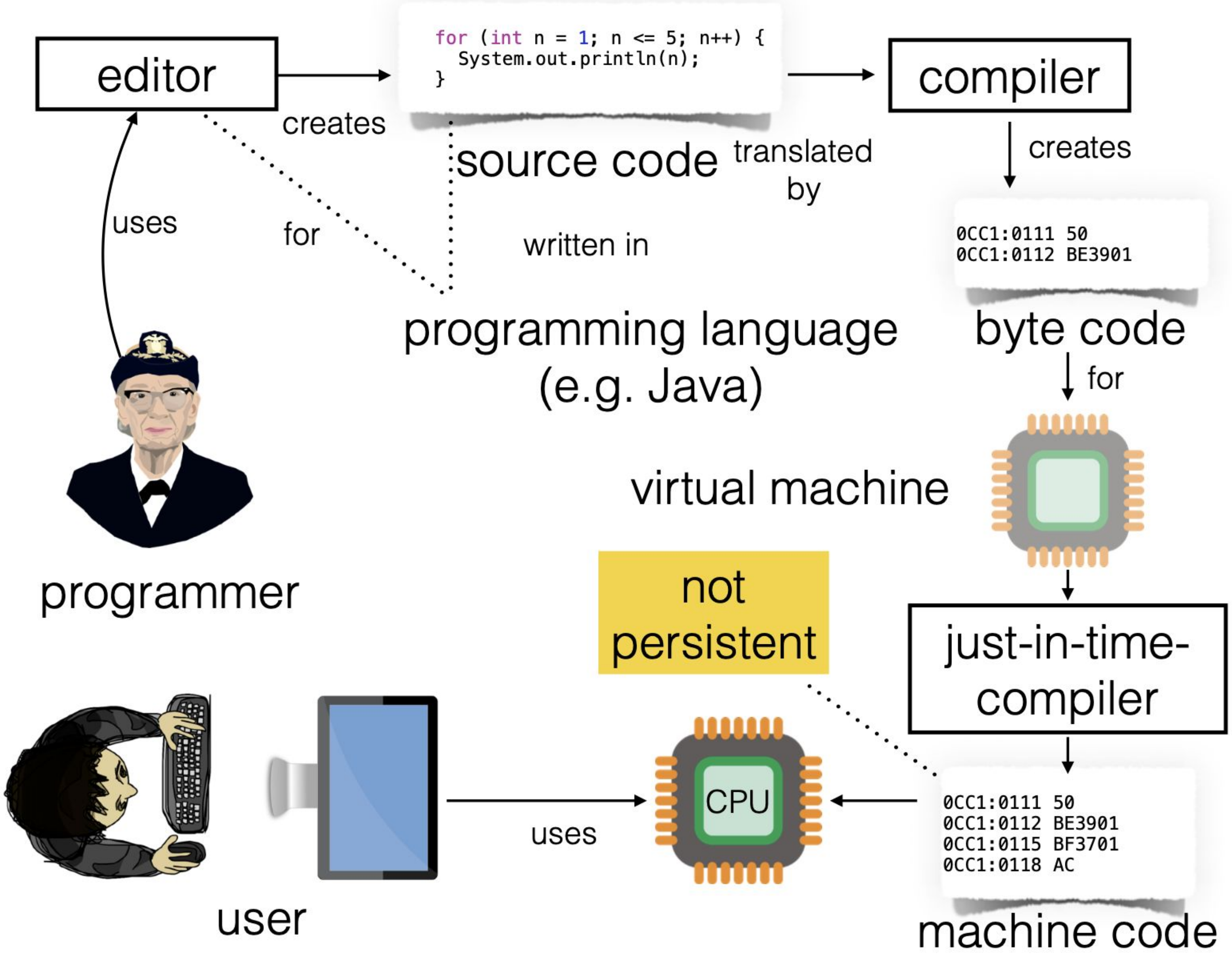
programmer

user

uses

CPU

Just in Time Compiler



editor

creates

```
for (int n = 1; n <= 5; n++) {
  System.out.println(n);
}
```

source code

translated by

compiler

creates

```
0CC1:0111 50
0CC1:0112 BE3901
```

byte code

for

virtual machine

just-in-time-compiler

```
0CC1:0111 50
0CC1:0112 BE3901
0CC1:0115 BF3701
0CC1:0118 AC
```

machine code

not persistent

uses

for

written in

programming language (e.g. Java)

programmer

user

uses

Interpreter vs Compiler

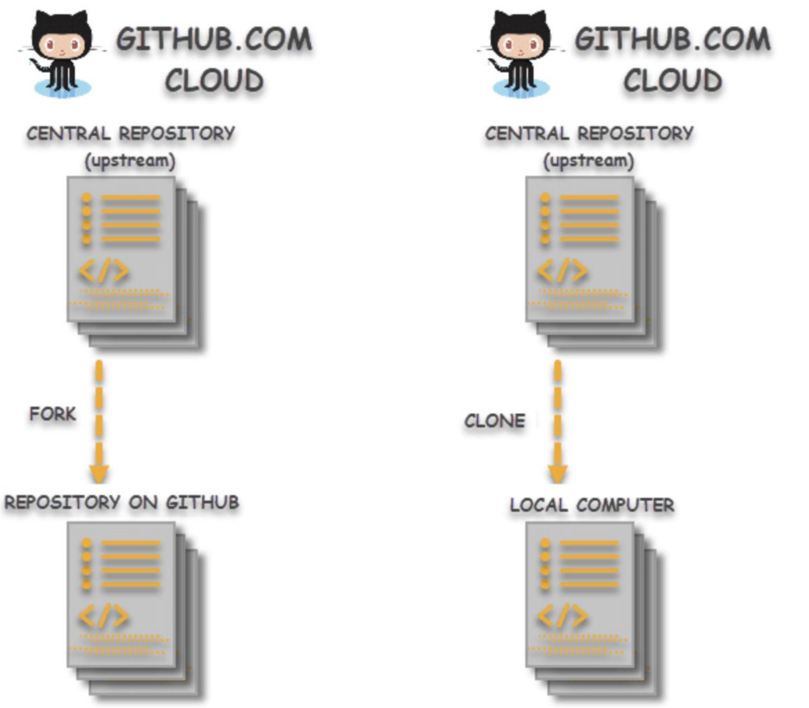
[6]

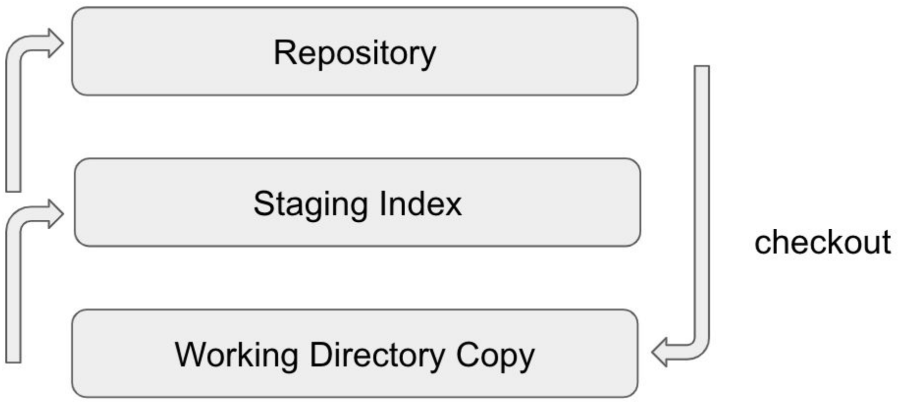
Interpreter	Compiler
Translates program one statement at a time.	Scans the entire program and translates it as a whole into machine code.
Interpreters usually take less amount of time to analyze the source code. However, the overall execution time is comparatively slower than compilers.	Compilers usually take a large amount of time to analyze the source code. However, the overall execution time is comparatively faster than interpreters.
No intermediate object code is generated, hence are memory efficient.	Generates intermediate object code which further requires linking, hence requires more memory.
Programming languages like JavaScript, Python, Ruby use interpreters.	Programming languages like C, C++, Java use compilers.

Git and GitHub

GitHub

- Where is the code?
  - Online repository
  - Locally on your machine
- Who can see a change?
  - Just you
  - Everyone
  - How about updates?
- When do I commit?
  - A unit of work has been completed
- So many commands to learn!
  - You really only need 4-5 and will reuse them over and over
  - Don't use IDE to do it for you because next semester you are on a different language (and IDE) already





Three Tree Architecture of Git

Summary

★ Lab Changes
★ Programming Background
★ Finish GitHub example from Wednesday

References

1. Kenny Eliason, Difference between OOP and Procedural Programming, https://neonbrand.com/website-design/procedural-programming-vs-object-oriented-programming-a-review/, 2013, Accessed Sep 2020.
2. The Real Python, 2012-2018, https://realpython.com/switching-to-python/, Accessed Sep 2020.
3. Learn Python in one day, Jamie Chan, 2014
4. Moutaz Haddara, Introduction to Object-oriented programming, slideshare, 2014.
5. Jamie Chan, Learn Python in one day, 2014.
6. Interpreter vs Compiler, https://www.programiz.com/article/difference-compiler-interpreter, accessed Sep 2020.