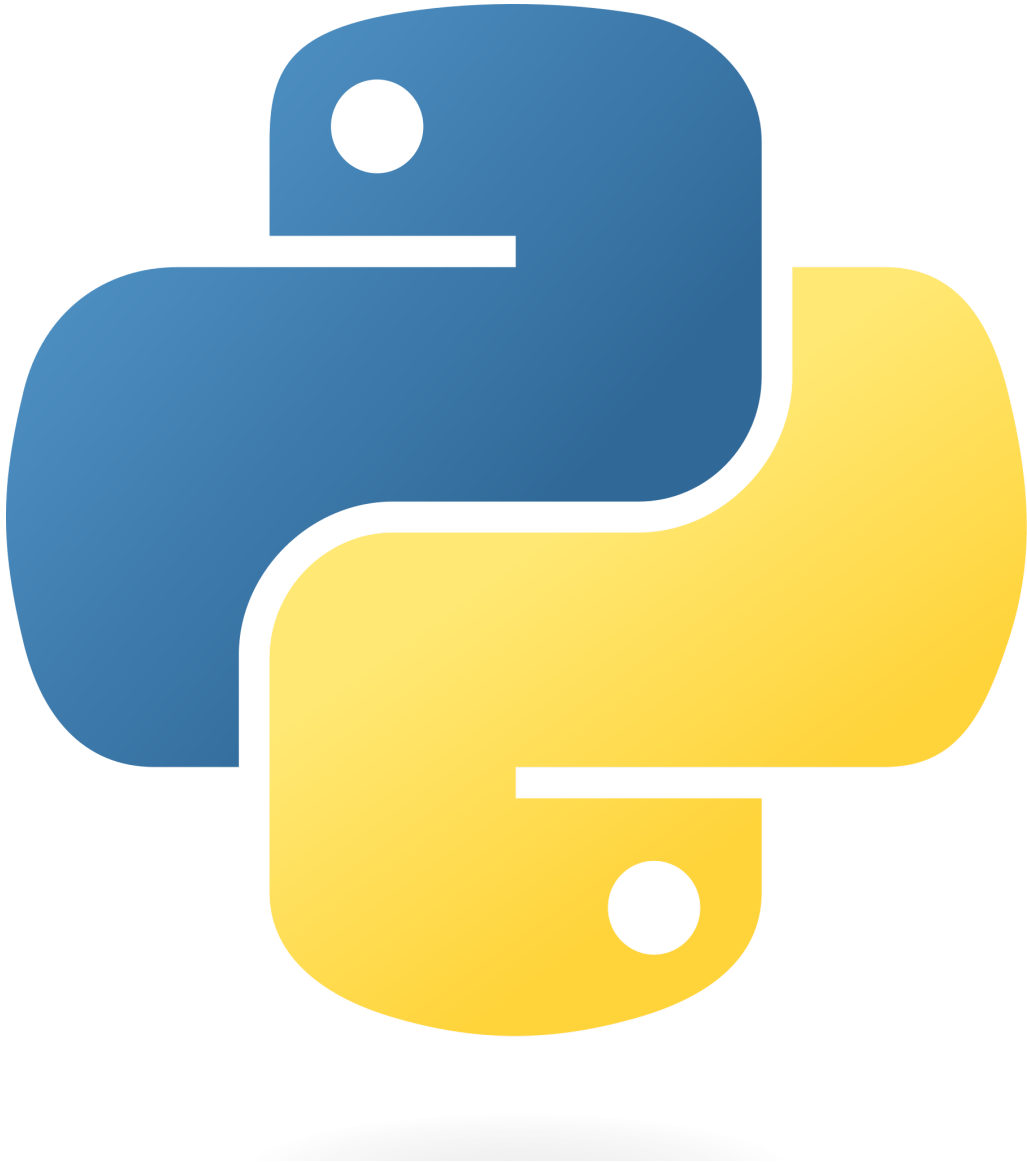
Coding for Beginners w/ Python



Attendance

A bit about myself. 👉

Zavaar Shah

- 3rd year of CS
- Pres SCD
- Pres GDSC
- VP ESS

What is programming/coding?

- Writing instructions for a computer to execute.
- Computers are dumb, but fast.
- Computers don't understand human languages.

What is Python?

- Python is a **high-level**, interpreted programming language.
- It is known for its simplicity and readability.
- It is a great language for beginners.

Let's get started!

Download Python: python.org

Download code editor: code.visualstudio.com

OR

Use an online editor (limited experience): online-python

Hello, World!

main.py

```
# This is a comment, they are ignored by the computer.
print("Hello, World!") # This is a print statement.
```

Running the code

- Open the terminal.
- Navigate to the folder where `main.py` is located.
- Run the command

```
$ python main.py
```

> It may be `py`, `python3`, `py3` on some systems.

Variables

```
a = 10
b = 5

c = a + b

print(a)
# what will this print?
```

```python
a = 10
b = 5

c = a + b

print(b)
# what will this print?
```

```python
a = 10
b = 5

c = a + b

print(c)
# what will this print?
```

Math operations

```
a = 10
b = 5

print("Sum:", a + b) # Sum: 15
print("Difference:", a - b) # Difference: 5
print("Product:", a * b) # Product: 50
print("Quotient: " + str(a / b)) # Quotient: 2.0
```

```
c = "Hello "
d = "World!"

print(c + d) # Hello World!
```

Data types

`int` - Integer number

```
a = 10
```

`float` - Floating point number

```
b = 2.5
```

`str` - String

```
c = "Hello, World!"
# or w/ single
quotes
```

`bool` - Boolean

```
d =
True #
or
False
```

Conditionals

```
name = "Zavaar"
age = 20

if age >= 18:
    print(name, "is an adult.")
else:
    print(name, "is a minor.")
    print("He will be an adult in " + str(18 - age) + " years.")

# Outputs: Zavaar is an adult.
```

Lists

`list` - Ordered collection of items

```
fruits = ["apple", "banana", "orange"]

print("the fruits are:", fruits)
# the fruits are: ['apple', 'banana', 'orange']


    print("the first fruit is:", fruits[0])
    print("the second fruit is:", fruits[1])
    print("the third fruit is:", fruits[2])
```

For loops

```
fruits = ["apple", "banana", "orange"]
for fruit in fruits:
    print(fruit)
# apple
# banana
# orange
```

```
for i in range(len(fruits)):
    print(i, fruits[i])
# 0 apple
# 1 banana
# 2 orange
```

While loops

```
i = 0
while i < 5:
    print(i)
    i += 1
# 0
# 1
# 2
# 3
# 4
```

Functions

```
# Define a function
def greet():
    print("Hello")


# Call the function
greet()


 # Outputs: Hello
```

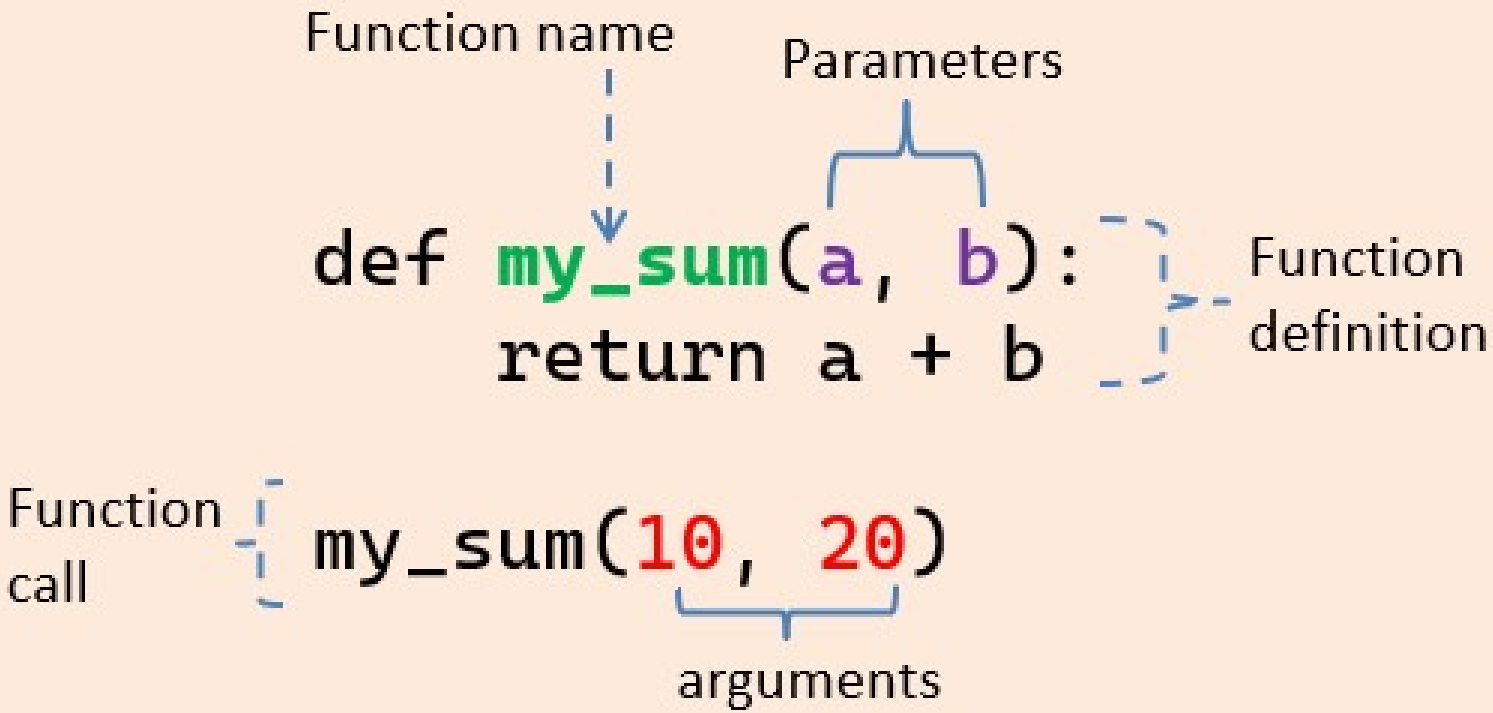
Functions with arguments

```
def greet(name):
    print("Hello", name)


greet("Zavaar")
a = "guy"
greet(a)


# Hello Zavaar
# Hello guy
```

Functions with return values

```
def sum(a, b):
    return a + b


result = sum(10, 5)

print(result) # 15
print(sum(9,10)) # 19
```

Functions overview



Parameters are mentioned in the function definition.
Actual parameters(arguments) are passed during a function call

Dictionaries

`dict` - Key-value pairs

```python
person = {
    "name": "Zavaar",
    "age": 20,
    "is_student": True
}
print(person["name"]) # Zavaar
if person["is_student"]:
    print("He is a student.")
else:
    print("He is not a student.")
# He is a student.
person["age"] = 21 # reassigning a value
print(person["age"]) # 21
```

There is a better way to interact with data.

With Object-oriented programming.

Classes

```
class Person:
    def __init__(self, name, age, is_student=False):
        self.name = name
        self.age = age
        self.is_student = is_student

    def greet(self):
        print("Hello", self.name)

    def is_adult(self):
        return self.age >= 18
```

Using the class

Instantiate the class

```
zavaar = Person("Zavaar", 20, True)
```

Now we can..

```
zavaar.greet() # Hello Zavaar
print(zavaar.is_adult()) # True
```

4 Concepts of OOP

I/O operations

```
# Reading from console input
name = input("Enter your name: ")
print("Hello", name)
```

```
# reading from a file
file = open("file.txt", "r")
print(file.read())
file.close()
```

```
# writing to a file
file = open("file.txt", "w")
file.write("Hello, World!\n")
file.close()
```

Let's build some programs.

BMI Calculator

Code can be found here.

```python
# BMI = 703 * ( lbs / (inches ^ 2))
def bmi(weight, height):
    return 703 * (weight / (height ** 2))

def bmi_classification(bmi):
    if bmi < 18.5:
        return "underweight"
    elif bmi < 25:
        return "normal"
    elif bmi < 30:
        return "overweight"
    else:
        return "obese"

weight = float(input("Enter your weight in lbs: "))
heightFt = float(input("Enter your height in feet: "))
heightIn = float(input("Enter the remaining height in inches: "))
# height = ( height in feet * 12 ) + extra inches
height = (heightFt * 12) + heightIn

your_bmi = bmi(weight, height)
print("Your BMI is:", math.floor(your_bmi))
print("You are", bmi_classification(your_bmi))
```

Rock, Paper, Scissors

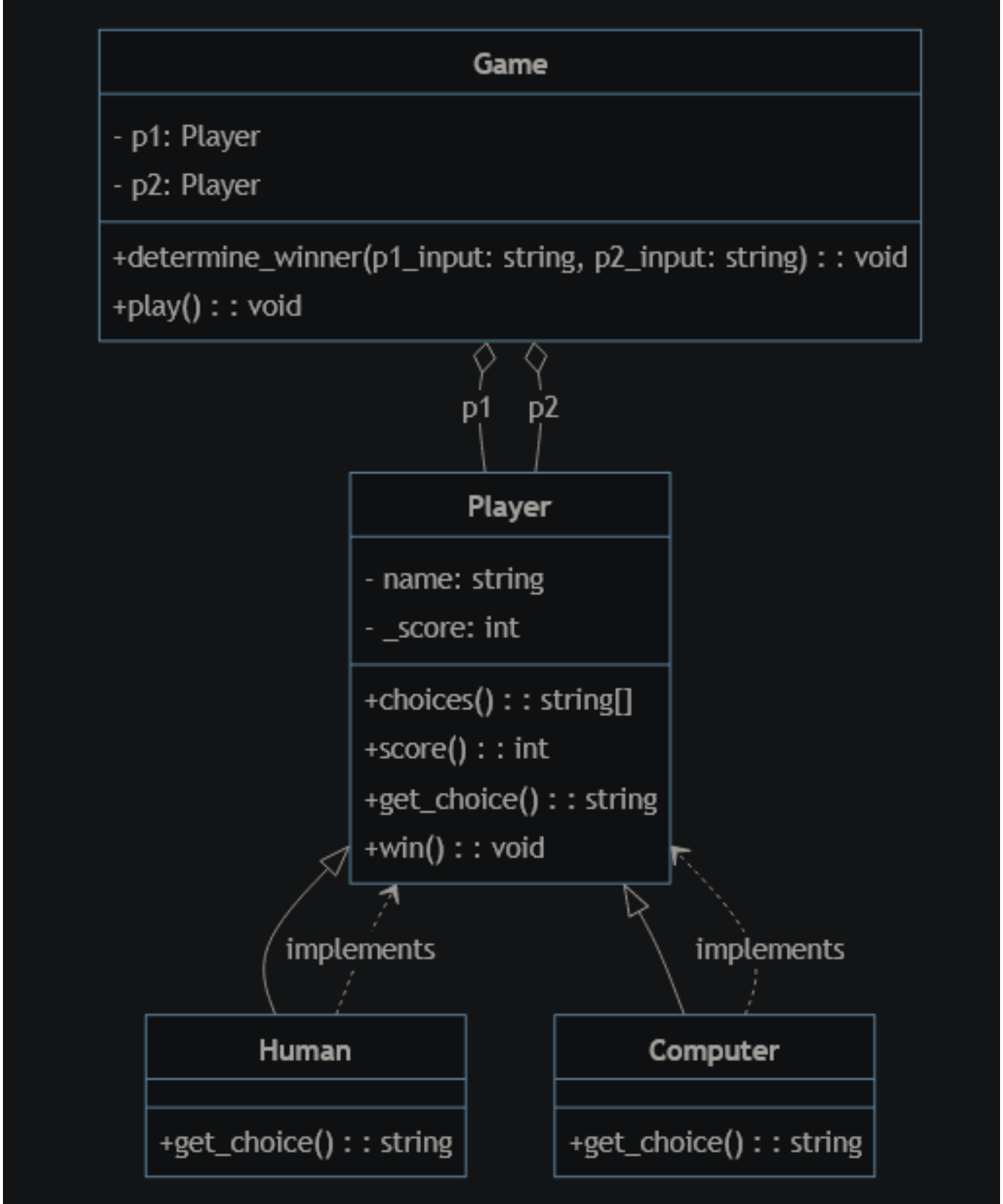
Code can be found here.

```python
import random

def get_user_choice():
    return input("Enter your choice: ").lower()

def get_computer_choice():
    return random.choice(["rock", "paper", "scissors"])

def determine_winner(user, computer):
    if user == computer:
        return "It's a tie!"
    if user == "rock":
        return "You win!" if computer == "scissors" else "You lose!"
    if user == "paper":
        return "You win!" if computer == "rock" else "You lose!"
    if user == "scissors":
        return "You win!" if computer == "paper" else "You lose!"

while True:
    user = get_user_choice()
    computer = get_computer_choice()
    print("Computer chose:", computer)
    print(determine_winner(user, computer))
    if input("Play again? (y/n): ").lower() != "y":
```

Rock, Paper, Scissors, with OOP

Code can be found here.

```python
import random
from abc import ABC, abstractmethod

class Player:
    def __init__(self, name):
        self.name = name
        self._score = 0

    @property
    def choices(self):
        return ["rock", "paper", "scissors"]

    @property
    def score(self):
        return self._score

    @abstractmethod
    def get_choice(self):
        pass

    def win(self):
        self._score += 1
        print(self.name, "wins!")
```


Game
- p1: Player
- p2: Player
+determine_winner(p1_input: string, p2_input: string) : : void
+play() : : void
p1
p2
Player
- name: string
- _score: int
+choices() : : string[]
+score() : : int
+get_choice() : : string
+win() : : void
implements
implements
Human
+get_choice() : : string
Computer
+get_choice() : : string

Thanks for attending!

Additional resources

Code

More recordings