**Madiha Aimon Tappal**

[**madihaaimon@gmail.com**](mailto:madihaaimon@gmail.com)

**Data Engineering Batch – 1**

**Day – 7 Assignment**

**Python**

Abstract class:

from abc import \*  
  
  
class Demo1(ABC):  
 @abstractmethod  
 def m1(self):  
 pass  
  
 @abstractmethod  
 def m2(self):  
 pass

Class:

class Person:  
 def \_\_init\_\_(self, name, age):  
 self.name = name  
 self.age = age  
  
 def print\_value(self):  
 print(f" name = {self.name}")  
 print(f" age = {self.age}")  
  
  
p1 = Person("John", 36)  
  
print(p1.name)  
print(p1.age)

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\class.py"

John

36

Process finished with exit code 0

Constructor: -

class Person:  
 def \_\_init\_\_(self, name, age):  
 self.name = name  
 self.age = age  
  
 def print\_value(self):  
 print(f" name = {self.name}")  
 print(f" age = {self.age}")  
  
  
p1 = Person("John", 36)  
  
print(p1.name)  
print(p1.age)

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\class\_constructor.py"

John

36

Process finished with exit code 0

Dictionary: -

def dictionary\_program():  
 student\_info = {  
 'name': 'John',  
 'age': 20,  
 'grade': 'A',  
 'courses': ['Math', 'English', 'Science']  
 }  
  
 # Displaying the original dictionary  
 print("Original Dictionary:", student\_info)  
  
 # Accessing values using keys  
 print("Name:", student\_info['name'])  
 print("Age:", student\_info['age'])  
  
 # Modifying a value in the dictionary  
 student\_info['age'] = 21  
  
 # Displaying the dictionary after modification  
 print("Dictionary after modification:", student\_info)  
  
 # Finding the number of key-value pairs in the dictionary  
 dictionary\_size = len(student\_info)  
 print("Number of key-value pairs in the dictionary:", dictionary\_size)  
  
  
dictionary\_program()

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\dictionary\_in\_python.py"

Original Dictionary: {'name': 'John', 'age': 20, 'grade': 'A', 'courses': ['Math', 'English', 'Science']}

Name: John

Age: 20

Dictionary after modification: {'name': 'John', 'age': 21, 'grade': 'A', 'courses': ['Math', 'English', 'Science']}

Number of key-value pairs in the dictionary: 4

Process finished with exit code 0

Exception handling: -

# exception handling  
x = 5  
y = "hello"  
try:  
 z = x + y  
except TypeError:  
 print("Error: cannot add an int and a str")

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\exception\_handling.py"

Error: cannot add an int and a str

Process finished with exit code 0

Hierarchical Inheritance: -

class Animal:  
 def speak(self):  
 print ("Animal speaks")  
  
class Dog (Animal):  
 def bark(self):  
 print ("Dog barks")  
  
class Cat (Animal):  
 def meow(self):  
 print ("Cat meows")  
  
dog1 = Dog ()  
dog1.speak()  
dog1.bark()

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\heirarchical\_inheritance.py"

Animal speaks

Dog barks

Process finished with exit code 0

Multi-level Inheritance: -

class Animal:  
 def speak(self):  
 print ("Animal speaks")  
  
  
class Dog (Animal):  
 def bark(self):  
 print ("Dog barks")  
  
  
class Puppy (Dog):  
 def wag\_tail(self):  
 print ("Puppy wags tail")  
  
  
puppy1 = Puppy()  
puppy1.speak()  
puppy1.bark()  
puppy1.wag\_tail()

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\multilevel\_inheritance.py"

Animal speaks

Dog barks

Puppy wags tail

Process finished with exit code 0

Multiple Inheritance: -

class Animal:  
 def speak(self):  
 print ("Animal speaks")  
  
  
class Pet:  
 def cuddle(self):  
 print ("Pet cuddles")  
  
  
class Dog (Animal, Pet):  
 def bark(self):  
 print ("Dog barks")  
  
  
dog1 = Dog ()  
dog1.speak()  
dog1.cuddle()  
dog1.bark()

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\multiple\_inheritance.py"

Animal speaks

Pet cuddles

Dog barks

Process finished with exit code

Singel -level inheritance: -

class Animal:  
 def speak(self):  
 print("Animal speaks")  
  
  
class Dog(Animal):  
 def bark(self):  
 print("Dog barks")  
  
  
dog1 = Dog()  
dog1.speak()  
dog1.bark()

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\single\_level\_inheritance.py"

Animal speaks

Dog barks

Process finished with exit code 0

If- Else: -

a = 33  
  
# if  
if a>23:  
 print("a")  
  
# if else  
b = 33  
if b > a:  
 print("b is greater than a")  
elif a == b:  
 print("a and b are equal")

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\if\_else.py"

a

a and b are equal

Process finished with exit code 0

Interface: -

from abc import ABC, abstractmethod  
  
  
class Shape (ABC):  
 @abstractmethod  
 def area(self):  
 pass  
  
  
class Circle (Shape):  
 def \_\_init\_\_ (self, radius):  
 self.radius = radius  
  
 def area(self):  
 return 3.14 \* self.radius \*\* 2  
  
  
circle = Circle(5)  
print("Area of the circle:", circle.area())

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\interface\_python.py"

Area of the circle: 78.5

Process finished with exit code 0

Lambda Function: -

def square ():  
 square = lambda x: x \*\* 2  
 print (square (5))  
  
  
square ()

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\lambda\_function.py"

25

Process finished with exit code 0

List -tuple: -

list\_num = [1,2,3,4]  
tup\_num = (1,2,3,4)  
  
  
list\_num [2] = 5  
print(list\_num)  
  
print(list\_num)  
print(tup\_num)  
  
print(type(list\_num))  
print(type(tup\_num))

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\list\_tuple.py"

[1, 2, 5, 4]

[1, 2, 5, 4]

(1, 2, 3, 4)

<class 'list'>

<class 'tuple'>

Process finished with exit code 0

Logical operators: -

def logical\_operators ():  
 age = int (input ("Enter your age: "))  
 citizen = input ("Are you a citizen? (yes/no): "). lower ()  
  
 if age >= 18 and citizen == 'yes':  
 print ("You are eligible to vote.")  
 else:  
 print ("You are not eligible to vote.")  
  
  
  
logical\_operators ()

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\logical\_operators.py"

Enter your age: 21

Are you a citizen? (yes/no): yes

You are eligible to vote.

Process finished with exit code 0

Loops: -

def while\_loop():  
 i = 1  
 while i < 6:  
 print(i)  
 if i == 3:  
 break  
 i += 1  
  
  
# pattern  
def for\_loop(n):  
 for i in range(0, n):  
  
 for j in range(0, i + 1):  
 print("\* ", end="")  
  
 # ending line after each row  
 print("\r")  
  
  
# break  
def break\_loop():  
 for i in range(5):  
 if i == 3:  
 break  
 print(i)  
  
  
# continue  
def continue\_loop():  
 for i in range(5):  
 if i == 3:  
 continue  
 print(i)  
  
  
# lambda  
def lambda\_function():  
 ans = lambda i: i + 10  
 print(ans(5))  
  
  
# string  
def string\_python():  
 a = " Hello, World! "  
 print(len(a))  
 print(a.upper())  
 print(a.lower())  
 print(a.title())  
 print(a.capitalize())  
 print(a.count("H"))  
 print(a.find("W"))  
  
  
  
n = 5  
for\_loop(n)  
while\_loop()  
break\_loop()  
continue\_loop()

Map reduce: -

data = [1, 2, 3, 4, 5]  
result1 = map (lambda x: x \* 2, data)  
   
  
result2 = filter (lambda x: x % 2 == 0, data)  
  
print(result1)  
print(result2)

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\map\_reduce.py"

<map object at 0x000002B65A201D80>

<filter object at 0x000002B65A201ED0>

Process finished with exit code 0

Method Overriding: -

class Parent:  
  
 def \_\_init\_\_(self):  
 self.value = "Inside Parent"  
  
 def show(self):  
 print(self.value)  
  
  
class Child (Parent):  
  
 def \_\_init\_\_(self):  
 self.value = "Inside Child"  
  
 def show(self):  
 print(self.value)  
  
  
obj1 = Parent ()  
obj2 = Child ()  
  
obj1.show()  
obj2.show()

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\method\_overriding.py"

Inside Parent

Inside Child

Process finished with exit code 0

Random Module: -

import random  
  
def guessing\_game ():  
 target\_number = random.randint (1, 10)  
  
 while True:  
 user\_guess = int(input("Guess the number between 1 and 10: "))  
  
 if user\_guess == target\_number:  
 print("Correct answer!")  
 break  
 elif user\_guess < target\_number:  
 print("Too small. Try again.")  
 else:  
 print ("Too large. Try again.")  
  
# Call the function  
guessing\_game ()

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\random\_module.py"

Guess the number between 1 and 10: 5

Too large. Try again.

Guess the number between 1 and 10: 1

Too small. Try again.

Guess the number between 1 and 10: 9

Too large. Try again.

Guess the number between 1 and 10: 2

Too small. Try again.

Guess the number between 1 and 10: 3

Too small. Try again.

Guess the number between 1 and 10: 4

Correct answer!

Process finished with exit code 0

Switch Case: -

def number\_to\_string(argument):  
 match argument:  
 case 0:  
 return "zero"  
 case 1:  
 return "one"  
 case 2:  
 return "two"  
 case default:  
 return "something"  
  
  
head = number\_to\_string(2)  
print(head)

C:\Users\madih\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\madih\Desktop\Data Engineering\Python Practice\switch\_case.py"

Two

Process finished with exit code 0