**Python**

✅ Topics Covered:  
-data type of python   
-python number  
-variable Assignments  
-strings  
-Indexing and slicing  
- String Properties and Methods  
- Print Formatting with Strings  
-Lists in Python  
-Dictionaries   
-Tuples  
-Sets   
-Booleans  
-Comparison operators   
-Chaining comparison operators  
-If , Elif and else statement   
-For loops   
-while loops   
-useful operators(Input, max,min)  
-list Comprehensions  
-Methods and the python documentation  
-def keyword  
-basic python Function  
-tuple unpacking with python function  
-interactions between python function  
-lambda expressions, map, filter

💬 Notes:  
- I think at least 30% of my course is complete , but I think I still have to revisit and refresh many thing for proper logic building like lambda, map, tuple unpacking many more miscellaneous stuff. But I will still going. Today I started to log my progress so I chose this day 1

**🗓️ Day 1 Saturday**

✅ Topics Covered:  
-OOP Introduction,   
-OOP Attributes and class keyword  
-  
**practice:**  
-make three mini project:  
1) **Basic Class**Create a class Book with attributes title, author, and pages.Add a method get\_info() that returns a string like:  
"Harry Potter by J.K. Rowling, 400 pages"  
2) Default Values  
Create a class Player with attributes:  
name (required)  
health (default: 100)  
mana (default: 50)  
Add a method status() that prints:  
"<name> has <health> health and <mana> mana."  
3) Class with Methods  
Create a class Car with:  
brand, model, and mileage  
Add:  
A method drive(distance) that adds to mileage  
A method info() that returns the current mileage

💬 Notes:  
-struggled with the third project where I have to give new value to a attribute but I got the hang of it

**🗓️ Day 2 Sunday**

✅ Topics Covered:  
-OOP Class object attributes and methods  
-OOP inheritance and Polymorphism

Practice:  
-Mini Combat System  
 create a class Character with:  
 Attribute: name, hp, attack\_power  
 Method: attack(other\_character) that reduces the other character’s HP

-Student records  
Create a class Student. Each student has:  
 name, and a marks list  
add:  
method average() that returns the average marks  
method topper(student\_list) that returns the name of the student with the highest average

Note- I have not practice, what I learn today I will do that tomorrow because those to challenges form the previous day take so much time. I have to say that I have to practice more lambda and list comprehension in the future.

**🗓️ Day 3 Monday**

✅ Topics Covered:   
-Didn’t learn anything new just practice

Practice:  
- Task 1: Animal Kingdom  
Create a base class Animal with:  
Attribute: name  
Method: speak() that returns "..." (generic sound)  
Then create:  
Dog class that inherits Animal, override speak() to return "Woof"  
Cat class that overrides it with "Meow"

- Task 2: Shape Area  
Create a base class Shape with a method:  
Then:  
Create a class Circle with radius  
Create a class Rectangle with length, breadth  
Override area() in both classes

- Task 3: Employee Hierarchy

Create a base class Employee with:  
name, salary  
method: display()  
Then a child class Manager that:  
Adds department  
Overrides display() using super().display() and adds more info

- Task 4: Unified Interface  
Make a function:  
def describe\_thing(obj):

obj.describe()  
Create classes:  
Fruit, Car, and Animal → each with a describe() method  
Call describe\_thing() on all of them — this is pure polymorphism in Python.

Note- I did pretty well today even though its new concept I completed to challenge pretty easily . I go bug in my coding when practicing but not that much and those bugs are pretty minor that can fix easily.

**🗓️ Day 4 Tuesday**

✅ Topics Covered:  
-OOP Special (magic/dunder) methods   
like \_\_str\_\_ ect thing  
-Fully completed OOP section

Practice:  
-Problem 1

Fill in the Line class methods to accept coordinates as a pair of tuples and return the slope and distance of the line.  
class Line:

def \_\_init\_\_(self,coor1,coor2):

self.coor1= coor1

self.coor2= coor2

def distance(self):

x,y=self.coor1

x1,y1=self.coor2

return ((x1-x)\*\*2+(y1-y)\*\*2)\*\*0.5

def slope(self):

x,y=self.coor1

x1,y1=self.coor2

return(y1-y)/(x1-x)

coordinate1 = (3,2)

coordinate2 = (8,10)

li = Line(coordinate1,coordinate2)

print(li.distance())

print(li.slope())

-problem 2  
class Cylinder:

def \_\_init\_\_(self,height=1,radius=1):

self.height=height

self.radius= radius

def volume(self):

return 3.14159\*(self.radius\*\*2)\*self.height

def surface\_area(self):

return 2\*3.14159\*self.radius\*self.height+2\*3.14159\*(self.radius\*\*2)

c = Cylinder(2,3)

print(c.volume())

print(c.surface\_area())

-and one small project   
I create a bank account class that has two attributes:   
For this challenge, create a bank account class that has two attributes:  
owner  
balance  
and two methods:  
deposit  
withdraw  
As an added requirement, withdrawals may not exceed the available balance.  
Instantiate your class, make several deposits and withdrawals, and test to make sure the account can't be overdrawn.  
  
note: done pretty fast I thought that I will get stuck in my mini project but I run through it pretty easily. And I think OOP will become my stronghold.

**🗓️ Day 5 Wednesday**

✅ Topics Covered:  
- Quick note on the PyPi website  
- Pip Install and PyPi  
- Modules and Packages  
- \_\_name\_\_ and "\_\_main\_\_"

Practice:  
- Challenge  
 1: Create Your Own Calculator Module  
Goal: Build a .py file that acts like a library.  
Make a file: mymath.py  
Add these functions:  
def add(x, y): return x + y  
def subtract(x, y): return x – y  
def multiply(x, y): return x \* y  
def divide(x, y): return x / y if y != 0 else "Cannot divide by zero"  
2.In a new file main.py, write:  
import mymath

print(mymath.add(10, 5))

print(mymath.divide(12, 0))  
  
- Challenge 2: Use from module import function

Now do:  
from mymath import add, multiply

print(add(2, 3))

print(multiply(4, 5))

- Challenge 3: Create a Package with \_\_init\_\_.py

Make a folder called tools  
Inside it, make two files: texttools.py and numbertools.py  
In texttools.py:  
def shout(text):

return text.upper() + "!"  
in numbertools.py:  
def square(n):

return n \* n

add and empty \_\_init\_\_.py file to tools  
in main.py  
  
- Install a PyPI Package and Use It

Install emoji or colorama from terminal:

Note- first when learning I feel so confused but after completing those challenges I can pretty easily handle module and packages. And I also completed the hole section

**🗓️ Day 6 Thursday**

✅ Topics Covered:  
-Errors and Exception Handling  
learn about try, except, finally  
  
Practice:  
- **Challenge 1: Safe Division (🌶️ Easy)**Ask the user for two numbers and divide them.  
If the second number is 0, catch the error and print a friendly message.  
Also use finally to print: "Program finished." no matter what.  
- **Challenge 2: Integer Input Only (🌶️🌶️ Medium)**Write a program that asks the user to enter **5 integers**.If the user types a non-integer (like “hello” or 5.5), catch the error and ask again without crashing.  
When done, print the list of all 5 valid integers.  
- **Challenge 3: File Reader from Hell (🌶️🌶️🌶️ Spicy)**Ask the user to input a filename. Try to open and read it.If the file doesn't exist, catch the error and say: "File not found."Use finally to print: "Attempted file read."Bonus: If the file is too large (>10MB), raise your own error.  
- **Banking Error Handler (🌶️🌶️🌶️🌶️🔥)**Build on your Bank Account class:

* deposit() should raise an error if user enters a negative amount
* withdraw() should raise an error if amount is negative or greater than balance
* Catch all errors using try-except inside a wrapper function called run\_transaction()
* Always print "Transaction attempted." using finally

Note:-first I am really happy to even come this far on my journey. So when I am trying to do the 1st challenge I tried to give my own twist using for loop. But I struggle with for loop in list iteration so I finally conquer that. Still struggling with individual error like typeError or exception and making custom error. That all

**🗓️ Day 7 Friday**

✅ Topics Covered:  
-learn about pylint   
-Running tests with the unittest library

Practice:  
-error and exceptions homework giving by the course teacher done it withing 30 mini 3 problem   
- Challenge 1 — String Utilities (🌶 Easy)  
Create a file string\_utils.py with:  
A function capitalize\_first\_letter(word) that capitalizes the first letter.  
A function is\_palindrome(word) that returns True if it’s a palindrome, False otherwise.  
Create test\_string\_utils.py with:  
unittest test cases for both functions.  
Use at least 3 test cases per function.  
Run Pylint on both files and fix all style issues until you get a score ≥ 9.5.

Note:  
- I am getting more confident in coding but oh my god testing is so fucking hard especially the unittesting with other test file importing remembering so many function name like TestCase. Oh my god but I did it and even get a 10 out 10 in with challenge 1 with pylint

**🗓️ Day 12 Thursday, date 14.08.2025**

✅ Topics Covered:  
-completed my warmup project call build a card game call war   
-also learn about git and how to use it basic no command yet only in vs code

practice:-  
-I completed war card game using class in vs code. I create card.py, player.py, deck.py, and game.py   
it was so fucking hard it took me two fucking day to completed it I always get stuck in a away I can’t even tell oh my god

note:- I couldn’t log for so many day its because I was very busy I did many thing in those un-logged day like reviewing my weakness like list comp, then lambda, map filter, Althouse thing I will take log from this day more seriously from now on

**🗓️ Day 13 Friday, date 15.08.2025**

✅ Topics Covered:  
-Start my second milestone project. Making blackjack from scratch.

Progress so far is that I already build the card and deck file in vs code struggling with hand file right now then I will be left with player file and game file where I will run the game   
  
Note:  
-logic building is so fucking hard even 40 day later and consistent coding I know its still short period of time frame but. Isn’t it I am still struggling with how to build logic with what I know if there was a part I don’t know what to do then I can accept that but even knowing all the basics I can’t still put together them and build something form them.

**🗓️ Day 14 Sunday, date 17.08.2025**

✅ Topics Covered:  
-Start my second milestone project. Making blackjack from scratch.  
completed the hand.py and player.py it was so fucking hard to build logic

Note:  
-main thing I noticing is that I am forgetting many things that I should have like I know I to slice and index an list right but in middle of the coding I got blank of my coding is not working like shit.  
you know I know good amount you inheritance and polymorphism I forget that also I don’t know what is happening

**🗓️ Day 16 Tuesday, date 19.08.2025**

✅ Topics Covered:  
completed my second milestone project the blackjack game. With file names are card.py, deck.py, hand.py, chips.py. player.py, game.py   
  
Note:  
-the main problem I face as always logic building and some unknown that I don’t know I just look up   
it took me 3 day to complete this   
  
so for majority of the part of the logic I build it and then I send it to you chatgpt for review then you told me what wrong then I correct them. Some of the logic are so hard I have to search it and then able to done it.

**🗓️ Day 17 Wednesday, date 20.08.2025**

✅ Topics Covered:  
-learn about decorators  
-learn how to properly log using logging method   
-get little it better in git

Practice:  
- challenge1: Hello Logger  
Create a decorator @logger that prints:  
Calling function: <name>  
before running the function.  
Apply it to 2–3 functions (greet(), add(), multiply()).

Challenge2:  
Make a decorator @timer that measures how long a function takes to run.  
@timer

def slow\_func():

time.sleep(2)

slow\_func()  
Output should say something like:  
Execution time: 2.0001 seconds.  
  
I completed all of this challenges   
  
Note:- learn something new today feeling very happy .  
but the teacher say that we won’t need this decorators in our day-day coding if I don’t go into web development. But I didn’t flater and practice this challenges   
tomorrow is generators i will learn about yield next