Agenda :—

→ Time → module ✓
→ fstring → ✓
→ surprise ✓
→ Decorators ✓

methods → class ✓
→ static
→ special

time.time() → seconds → Jan 1, 1970

time.time() → 10 AM IST
→ total_1

time.time() → 10:02 AM IST
→ total_2

(1) How much time a function/method takes to execute.

(2) print("Error: ______") → Terminal
print("Database entry done for name: Manal") → Terminal
→ file operation → .txt

[time] → Error: " ______ " → txt
→ 30 days

CV

$$\begin{tikzcd}
 & & & & \text{A1} & \\
 & & & & \text{A2} & \\
\text{Python} \arrow[r] & \text{Tool} \arrow[r] & \text{DS} \arrow[r, leftrightarrow] \arrow[d, leftrightarrow] & \text{ML} \arrow[uur] \arrow[ur] \arrow[r] \arrow[dr] \arrow[ddr] & \text{A3} & \\
 & & \text{Statistics} & & \text{A9} & \\
 & & \text{Maths} & & \text{Neural Network} \arrow[r] \arrow[d] & \text{Human Brain} \\
 & & & & \text{Deep learning} \arrow[dl] \arrow[d] \arrow[dr] & \\
 & & & \text{Computer vision} & \text{Natural language processing} & \text{Reinforcement learning}
\end{tikzcd}$$

Photos, chatgpt, Ghibli

$$\begin{tikzcd}
\text{Human} \arrow[r] \arrow[dr] \arrow[ddr] & \text{Eyes} \\
 & \text{Ear} \\
 & \text{Mouth}
\end{tikzcd}$$

$$\begin{tikzcd}
 & \underline{\text{Specialization}} \arrow[r] & \text{DL} \arrow[r] & \boxed{\text{15-20 L}} \arrow[d] \\
\text{} & \underline{\text{Computer vision}} \arrow[l] \arrow[r] & \text{} & \underline{\text{Imp}}
\end{tikzcd}$$

Read text, understand

NLP

Decorators

→ function that takes another function as input & returns a new function.

New function will have added functionality without modifying the original function's code.

```
Class ChatGPT:
    def __init__(self):

    def research(self):

    def qA(self):
```

obj → dental health

Chatgpt → 2 second (QA)
→ 10$

Chat gpt → 50 minutes (Research)
→ 500 $

Q. Analyse → which function is being used the most → Answer

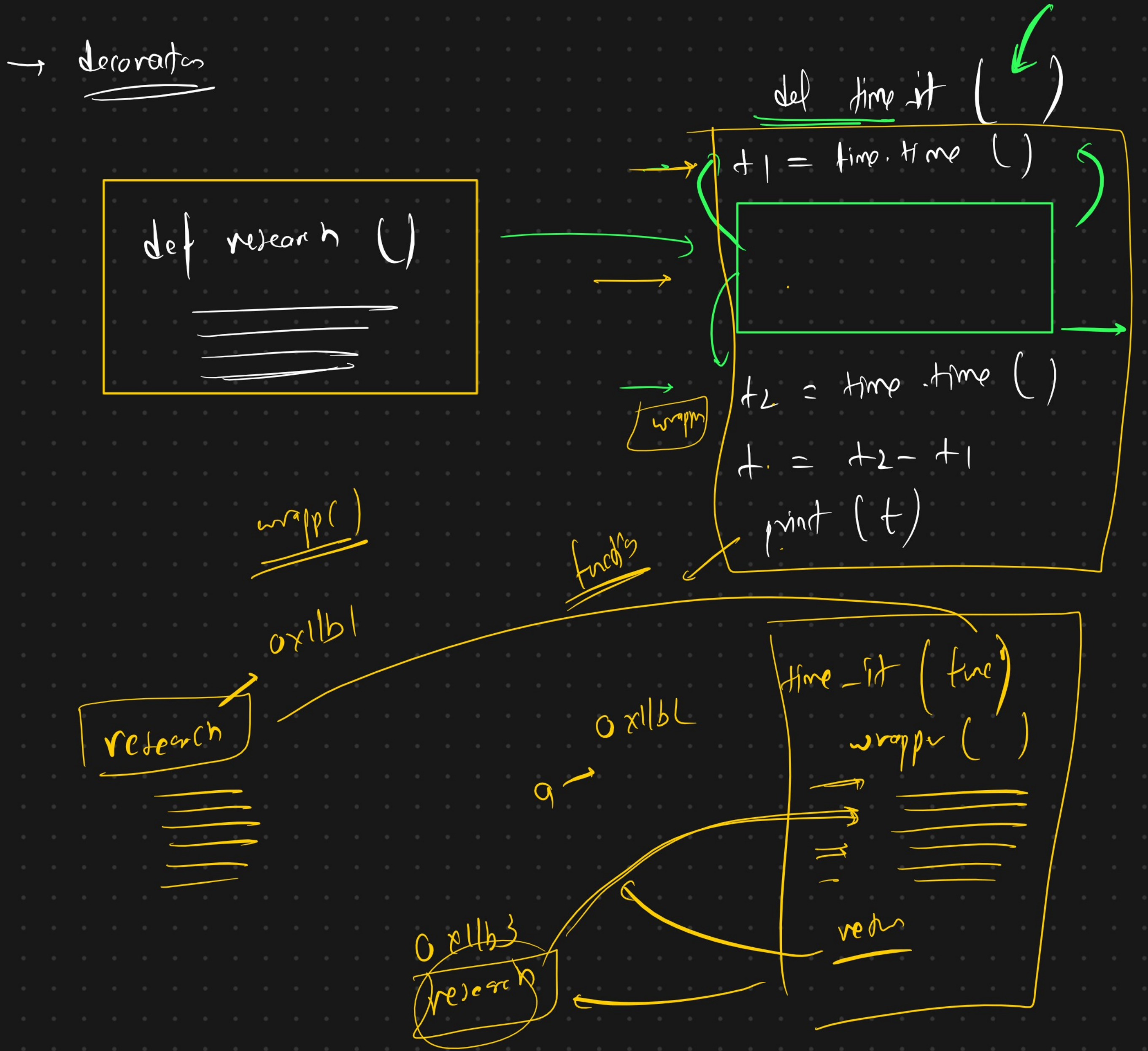
How much money it is buring.

```
def rese ( )
    t1 = time.time ( )        → 100's

    t2 = time.time ( )
    t = t2 - t1 → seconds

return research , t  → execution time
```

→ decorators

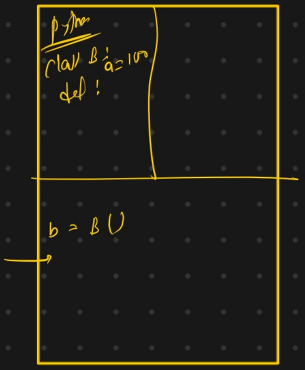


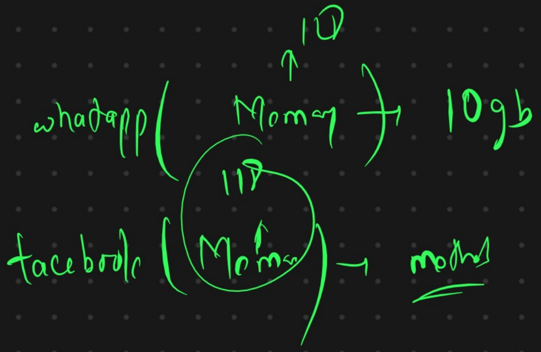
Class Memory :

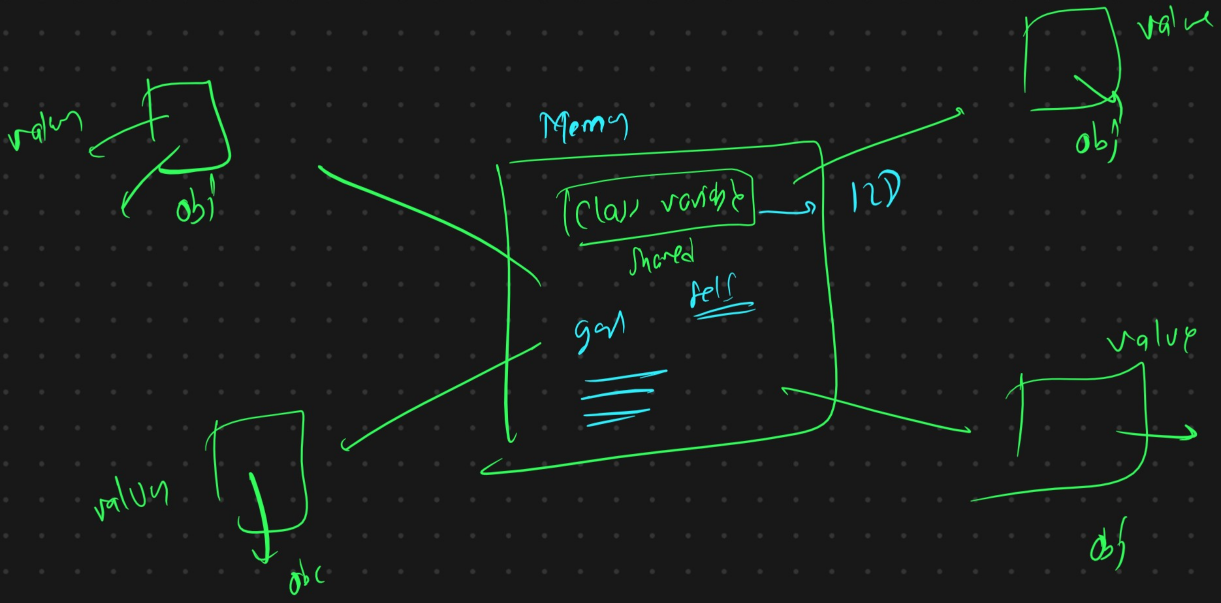
total_mem = 12D ⟶

def use_memory (self)

self.use = ______






value
obj
Memory
Class variable
12D
shared
self
gar
value
obj
valun
obc
value
obj