python Day-13 (Beginner)

Topic - Debugging : How to find & fix errors in your Code -

(*) why Debugging Matters

- Bugs are a natural part of programming - Even Experienced developers debug daily
- Rather than getting frustrated, Day 13 helps you develop a methodical approach to solving problems
- Debugging teaches deep understanding of how python executes Code.

Core Debugging Strategy - Key Steps

(1) Describe the problem
write down what the Code is supposed to do (vs) what it's actually doing
This helps clarify the exact issue insted of guessing

Example

```
for i in range (1,20):
    if i==20:
        print ("you got it")
```

Here the bug is that the loop never reaches 20
why? Because range (1,20) stops at 19

(2) Reproduce the Bug
Make the bug happen consistently if you cannot reproduce it reliably, you can't fix it confidently.
Try simple test runs or fixed inputs to isolate to problem

(3) Play Computer - line-by-line Evaluation
Pretend you are the python interpreter:
- Go through the code one line at a time
- Track variable values as they change
- This reveals logic errors before running code.

This is one of the most effective debugging habits.

(4) Pay Attention to Errors & IDE warnings
- Python error messages (tracebacks) are extremely helpful
- Many editors show red underlines for syntax or obvious issues.
- Dont ignore these hints - they often point you directly to the bug.

(5) Use print() statments strategically
inserting debug prints lets you inspect variables at runtime

```
year = int(input("what's your year of birth?"))
if year >1980 and year<1994:
    print("Debug:", year, "→ Millennial")
```

print statments help you see what python is doing inside your code.

(6) Handle Errors with try/Except
for user inputs and other risky Operations, wrap code to avoid Crashes:

```
try:
    age = int(input("How old are you?"))
Except ValueError:
    print("Invalid input - please enter a no.")
```

This prevents the program from stopping Unexpectedly

(7) Use a Debugger
A debugger tool lets you:
- Set breakpoints
- Step through code line by line
- Inspect variables and Call stack

This is powerful for Complex programs where print statments aren't enough

(8) Step Away when Stuck
Sometimes stepping back and returning later refreshes your perspective.
Even Experienced developers uses this trick to crack through bugs!

(9) Test Often, Don't write to much at Once
Run your Code frequently - don't write 50 lines then test
This way you catch errors early and make debugging Simpler

(10) Ask for Help
if Stuck don't hesitate to
- Talk to a peer
- Search Online (eg. Stack Overflow)
- Use course discussion forums

A fresh perspective Often reveals Simple fixes. :)