



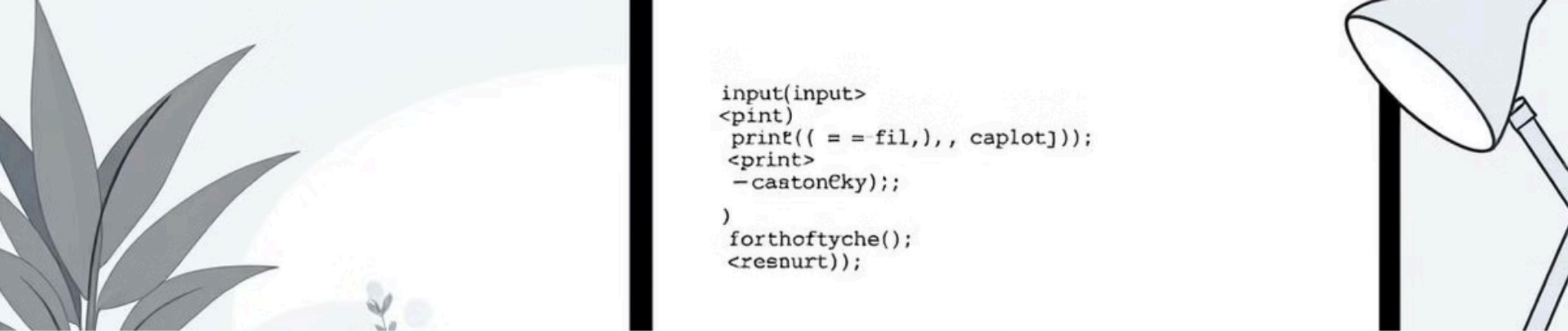
User Input and Output Basics in Python

Day 7 of 75 Days Coding Challenge.

Master Python's interactive capabilities.



by Aravind K



```
input(input>
<pint>
print(( == fil,, caplot));
<print>
-castoneky);
)
forthoftyche();
<resurt>);
```

Introduction to User Input and Output

1 Input()

Capture user data effortlessly.

2 Print()

Display information clearly and concisely.

3 F-strings

Format output for enhanced readability.

4 Error Handling

Gracefully manage unexpected user inputs.



Taking Input from Users

- 1 String Input
Capture text with `input()` function.
- 2 Numeric Input
Convert strings to numbers using `int()` or `float()`.
- 3 Multiple Inputs
Gather various data types in sequence.

Displaying Output with print()

Basic Syntax

`print()` displays values to console.

Multiple Arguments

Separate values with commas for easy formatting.

Customization

Modify separators and line endings for precise output.

String Formatting with f-strings

```
feetpree-e0... >
    batates fill, rodent, coston,
    Mateline (ft, the firrent.
    Vatalibee 11) esment,
    Tapercaste alire, irtetent,
    batorlbesfit, to fobceltbrohtping,
    Raterible it, lout, reimenated.<
    Vatorible in, to, ft, inbo tn taltt,
    Viyatlbeles, incouel,
    Watortbee thi roe-ic, respose()

Rebrite fill, instome teste,
<<Euntheales, talit,
<tystibeles, f..t.
Uatitanbles (biit.filt Pictigurg.
Bettorre aire, to (outfoe,/virementers,irclls,
Beteling fit, toser fritthomeschittl,
Matoribe ill; ratlof struigg,
Matertble ft, tuira, srsud,;
Mocaribee itt. bitc./frtatoner.bitty,
Matoribe fit, toses yester self)
Matortbellt.- toset (frtterbour fill, irc-isall)
Vatabibes:(fral irtt, frtyter scctes,
Watoribee for the it. tacuct;
eod.

Astoltables - asset fratatning;
Vatattbele slures (istericoAbett.(rait i.li))

Matoribee fill >
```

Syntax

Prefix string with 'f' and use {}.

Variable Insertion

Place variables directly inside curly braces.

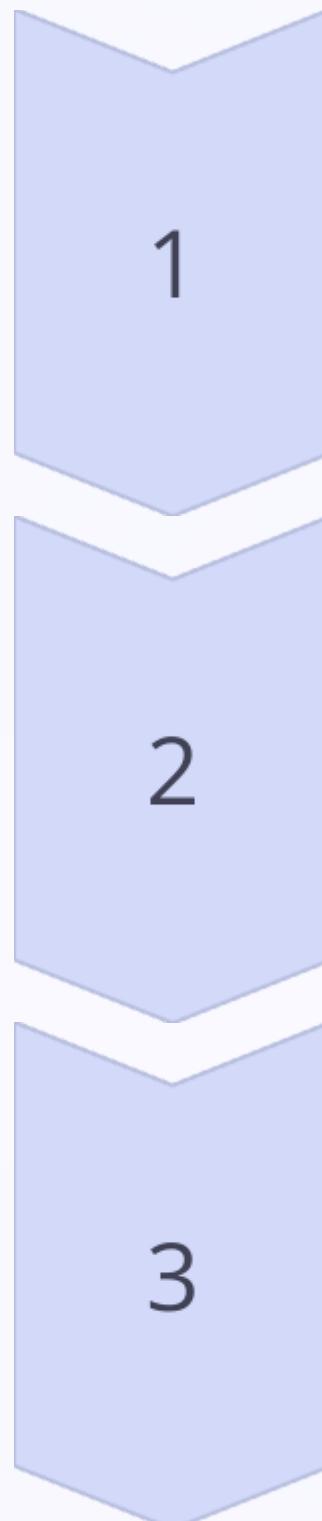
Expressions

Evaluate expressions within f-string brackets.

Formatting Options

Specify alignment, padding, and precision easily.

Handling Input Errors



Try Block

Attempt to execute potentially error-prone code.

Except Block

Catch and handle specific exceptions gracefully.

User Feedback

Provide clear error messages for better interaction.

```
1  expect(Ty-apen) {
2    exerttry larmme) {
3      ntp: isxetTy = >-> }
4      pan: tixt
5      you: tixt = <;
6      par: pld = soll-am)
7      iss tades adlaa>
8
9
10
11
12
13
14
15
```



Putting It All Together

```
! ladis python
'
Python: 72025.118:
(at the (bab 62llste: lo)
Prestive capeet: "ttage: lo
incessied to tero.11)
)
sizeef F7p.12):
)
tapoe:
Crpee: oucīting (apinge lo)
tratte graeming: lo)
Liak arctertons ('witer: swice
Ligh the Crater (Tapperr-sneplide
Lisk the Corter (faatte: snate
Lisk the Crater ("taper-warte: cucteasewipe: )
Liak ceriotsrioer (onallag)
Lisk tm recater (lor -ditentcreating)
Liak terippisetont: sent-/nolleed
Lisk crnacalds for -wvermtlle (harcpling)
Lisk the Covter- scceet (oction- edilep:
Stan teripytlat: ( repkige
Liak periguutet (retenving)
```

Input	Processing	Output
Name (string)	Store as-is	Formatted greeting
Age (integer)	Convert and validate	Age confirmation
Invalid input	Error handling	Friendly error message



Conclusion



User Input

Capture data effectively with `input()`.



Output Display

Present information clearly using `print()`.



Formatting

Enhance readability with f-strings.



Error Handling

Ensure smooth execution with `try-except`.