SE Bootcamp

Hyperiondev

Defensive Programming I – Error Handling

Objectives

- Discover the different types of errors that could occur in your programs and how to handle them.

Everyone Makes Mistakes

- ★ No programmer is perfect, and we’re going to make a lot of mistakes in our journey – and that is perfectly okay!

- ★ What separates the good programmers from the average ones is the ability to find and debug errors they encounter.

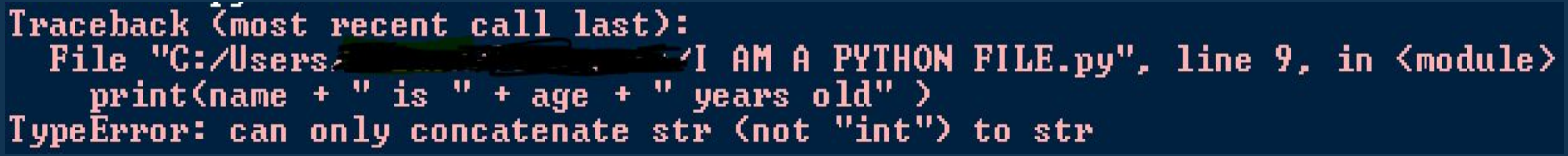
Error Messages

★ The output window of your IDE will usually show any and all Error messages if an error or mistake is detected.

★ It should display the type of error found as well as the line number in your code where the error occurred.

★ Your program will stop running immediately when an error is found.

Error Message Example



```
Traceback (most recent call last):
  File "C:/Users/[redacted]/I AM A PYTHON FILE.py", line 9, in <module>
    print(name + " is " + age + " years old" )
TypeError: can only concatenate str (not "int") to str
```

★ Looking at the above example:
  - The message states that the error occurred around line 9 – a good starting point for debugging.
  - It also states the type of error, which appears to be a TypeError. Useful, since we already could have ideas on how to fix the error.

Hyperiondev

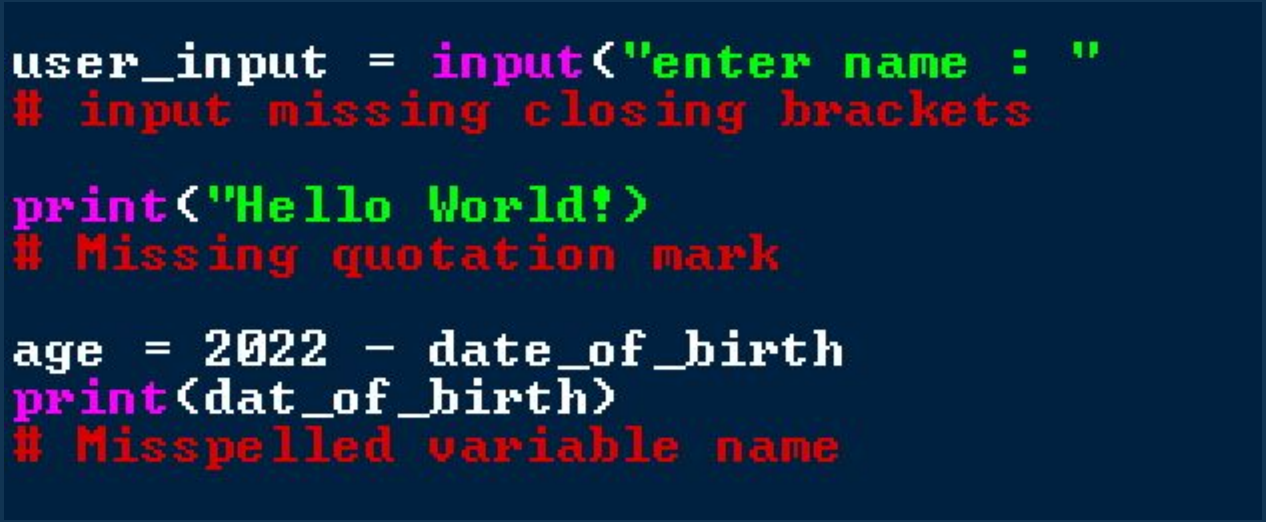
Syntax Errors

- ★ Some of the easiest errors to fix …
  - … Usually

- ★ Mainly caused by typos in code or Python specific keywords that were misspelled or rules that were not followed.

- ★ When incorrect syntax is detected, Python will stop running and display an error message.

Syntax Error Example



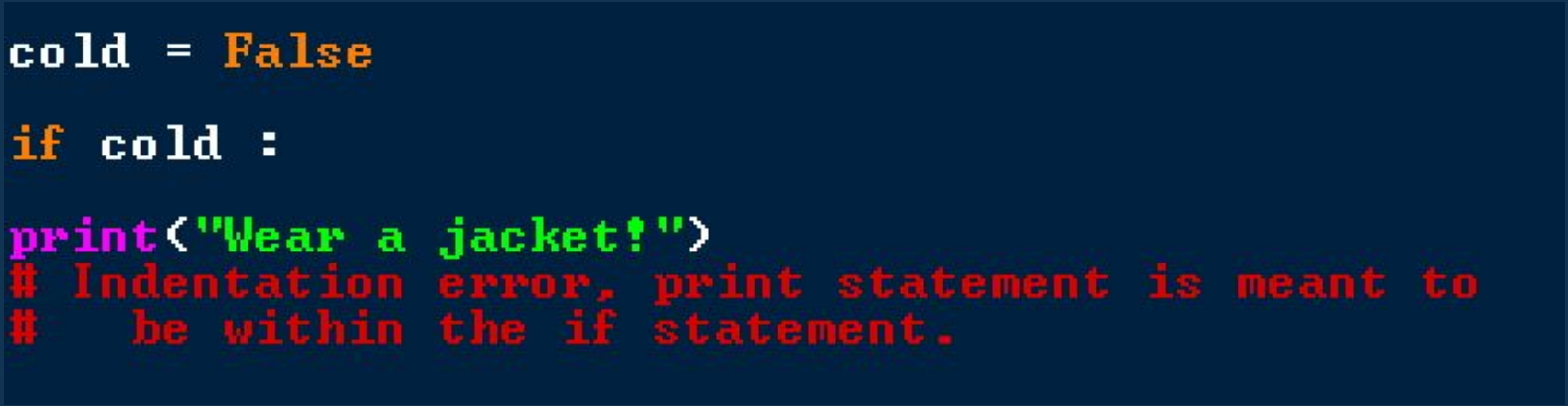
```
user_input = input("enter name : "
# input missing closing brackets

print("Hello World!)
# Missing quotation mark

age = 2022 - date_of_birth
print(dat_of_birth)
# Misspelled variable name
```

Hyperiondev

Indentation Errors

- ★ Indentation is important in programming.
- ★ Python uses indentation to understand where blocks of code start and stop.
- ★ The presence of indentation errors means that there is something wrong with the structure of the code.
- ★ A good rule of thumb: if a line of code ends with a colon (:), the next line should be indented.

Indentation Error Example



```
cold = False

if cold :

print("Wear a jacket!")
# Indentation error, print statement is meant to
#    be within the if statement.
```

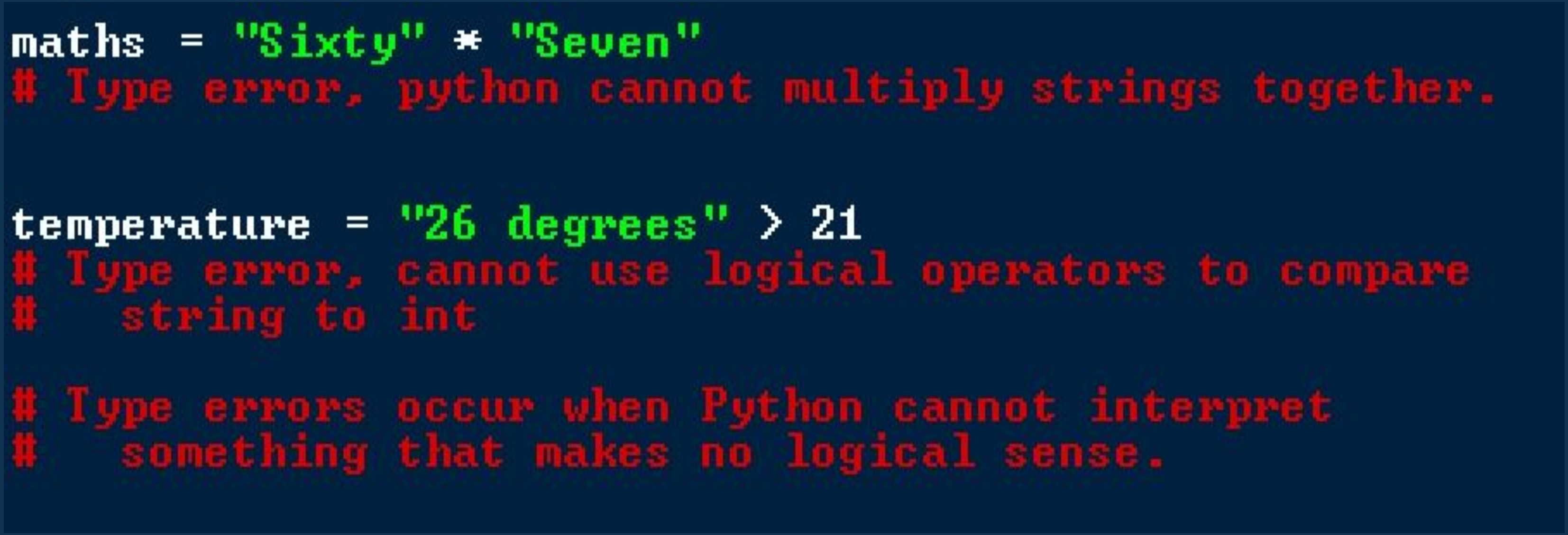
Type Errors

- ★ A type error occurs when your code has misinterpreted one type of data for another, like integers for strings.

- ★ Remember that for Python to actually work, your code needs to make logical sense so that Python can interpret it correctly and achieve the desired output.

Hyperiondev

Type Error Example



```python
maths = "Sixty" * "Seven"
# Type error, python cannot multiply strings together.


temperature = "26 degrees" > 21
# Type error, cannot use logical operators to compare
#   string to int

# Type errors occur when Python cannot interpret
#   something that makes no logical sense.
```

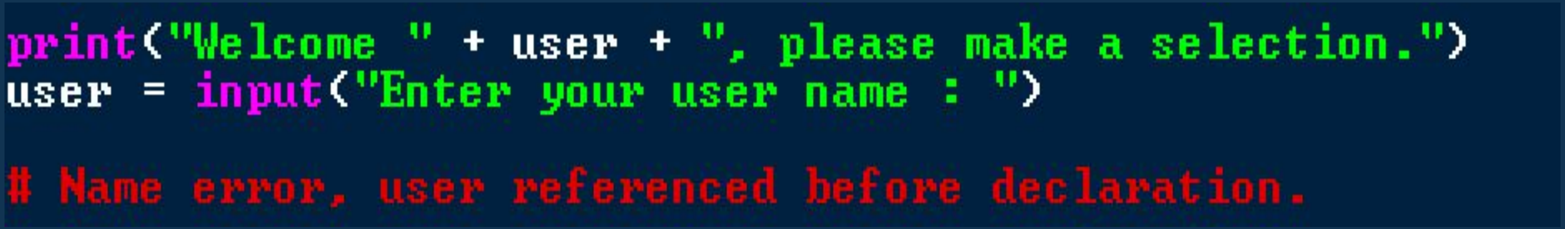
Hyperiondev

Name Errors

- ★ Naming errors occur when you try to reference or call a variable that has not been declared / created yet.

- ★ A good habit to get into when coding is to first define all variables, functions, etc. at the top of your program.

Name Error Example



```
print("Welcome " + user + ", please make a selection.")
user = input("Enter your user name : ")

# Name error, user referenced before declaration.
```

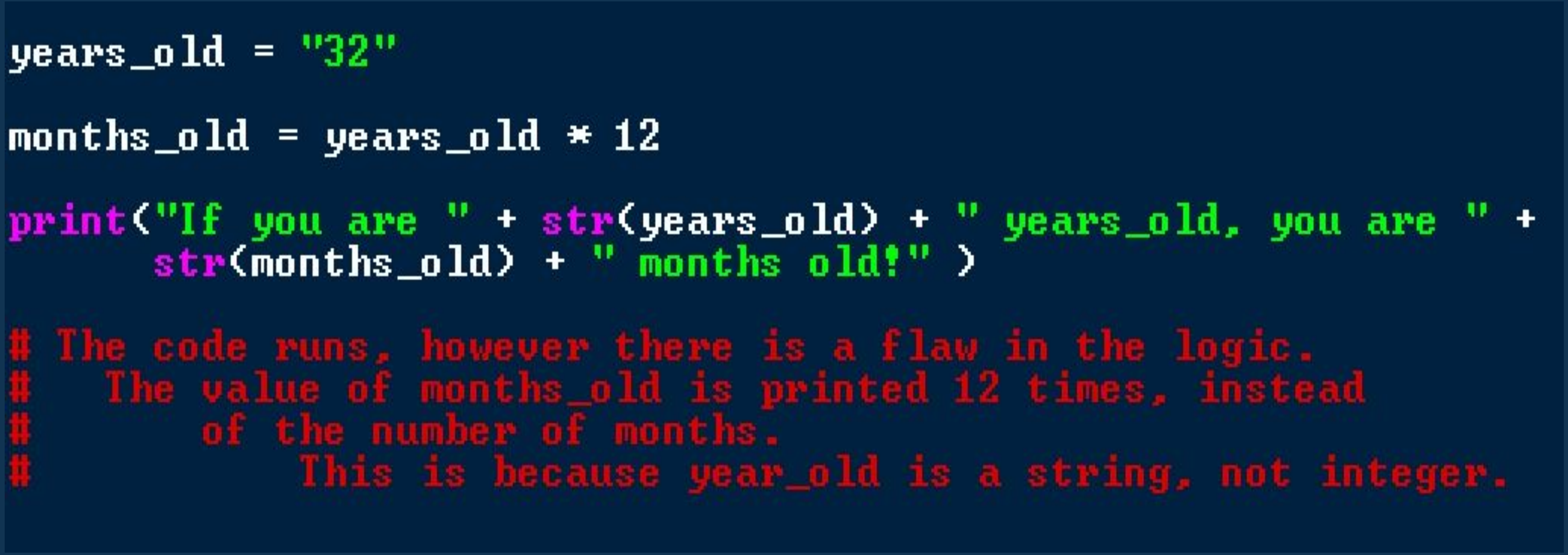
Hyperiondev

Logical Errors

★ Logical errors occur when your program is running, but the output you are receiving is not what you are expecting.

★ The code could be typed incorrectly, or perhaps an important line has been omitted, or the instructions given to the program have been coded in the wrong order.

Logical Error Example



```
years_old = "32"

months_old = years_old * 12

print("If you are " + str(years_old) + " years_old, you are " +
      str(months_old) + " months old!" )

# The code runs, however there is a flaw in the logic.
#   The value of months_old is printed 12 times, instead
#       of the number of months.
#           This is because year_old is a string, not integer.
```

Resources

Python website

https://www.python.org

Error handling

https://peps.python.org/pep-0498/#error-handling

F-strings

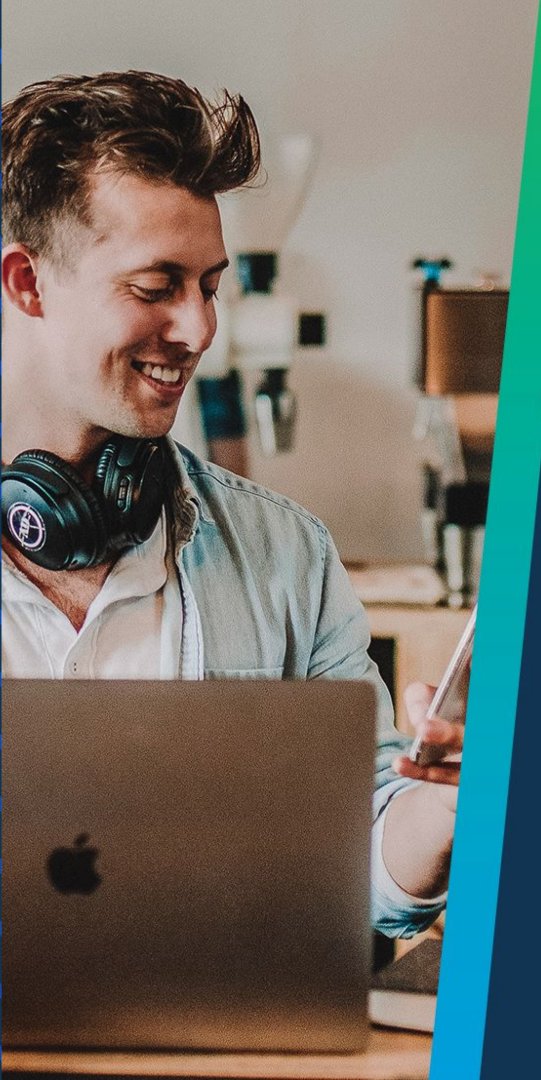
https://peps.python.org/pep-0498/

Hyperiondev

Hyperiondev

Q & A Section

Please use this time to ask any questions relating to the topic, should you have any.



Hyperiondev

Thank You for Joining Us