7. Problem Solving

How to approach programming issues

Error Types

Error types

- An **error** refers to a **deviation** from the **expected behavior** of a program.
- Errors can occur at different stages of the program's lifecycle, from writing the code to running it.
- Errors in Python can be categorized into **three main types:**

Syntax Errors

Logical Errors

Runtime Errors

Syntax Errors

- Syntax errors are the most common type of error in Python.
- They occur when the code is not written according to the **rules of** the Python **language**.
- They are usually detected by the Python interpreter and prevent the code from being executed.
- Examples of syntax errors include missing parentheses, incorrect indentation, and misspelled keywords.

Syntax Errors

Where are the bugs here?

```
print(“Hello, world!)

print(Hello, world!)

print(“Hello, world”)!
```

NOTE

“Bugs” are errors in program code

Syntax Errors

Where are the bugs here?

```
print(“Hello, world!)

print(Hello, world!)

print(“Hello, world”)!
```

NOTE

“Bugs” are errors in program code

Runtime Errors

- Runtime errors occur **during** the **execution** of the code.
- They are often caused by invalid input or other issues that **cannot be detected** by the Python interpreter.
- Examples of runtime errors include division by zero, accessing an index that does not exist in a list, and trying to open a file that does not exist.

Runtime Errors

Where’s the bug here?

```
x = 10

y = 0

z = x / y

print(z)
```

Runtime Errors

Where’s the bug here?

```
x = 10

y = 0

z = x / y

print(z)
```

The code will execute but it will throw a “Zero Division” error!

Runtime Errors

Where’s the bug here?

```
myList = [1, 2, 3]

print(f”First element: {myList[0]}”)

print(f”Last element: {myList[4]}”)
```

Runtime Errors

Where’s the bug here?

```
myList = [1, 2, 3]

print(f”First element: {myList[0]}”)

print(f”Last element: {myList[4]}”)
```

The code will execute but it will throw a “Index” error!

Logical Errors

- Logical errors occur when the code runs without “errors” but produces **incorrect results**.
- They are often caused by a mistake in the logic of the code.
- Examples of logical errors include using the wrong formula in a calculation, using the wrong variable in a loop, and using the wrong condition in an if statement.

Logical Errors

Where’s the bug here?

```
a = 10

b = 5

summation = a - b

print(summation)
```

Logical Errors

Where’s the bug here?

```
a = 10
b = 5
summation = a - b
print(summation)
```

Wrong operation! The code will run properly but it will give you the wrong answer!

Logical Errors

Where’s the bug here?

```
num1 = 10

num2 = 5

average = (num1 + num2) // 2

print(average)
```

Logical Errors

Where’s the bug here?

```
num1 = 10
num2 = 5
average = (num1 + num2) // 2
print(average)
```

Wrong operation! The code will run properly but it will give you the wrong answer!

Divide and conquer

Tackling complex problems

Often a programmer is faced with:

- Simple issues that are complex to solve (e.g. “bring me a cup of tea”)
- Complex issues that are complex to solve (e.g. “predict election outcome”)

To not get overwhelmed by the **complex solution** that needs to be implement, **Divide and conquer** the problem statement.

The goal: translate one **big problem** (task) into many **small problems** (steps)

As the exercises get **trickier**, you need to **plan ahead** better!

Tackling complex problems

Example: Check if income correlates with happiness?

1. Divide

Get income data

Get happiness data

Get correlation matrix

Get coefficient from matrix

Report coefficient

2. Conquer

```
incomeData = np.load('incomeDataset.npz')

happinessData = np.load('happinessData.npz')


correlation_matrix = np.corrcoef(dataset1, dataset2)


coef = correlation_matrix[0,1]


print(f"Correlation coefficient: {coef:.2f}")
```

3. Combine

```
import numpy as np
incomeData = np.load('incomeDataset.npz')
happinessData = np.load('happinessData.npz')
correlation_matrix = np.corrcoef(incomeData, happinessData)[0,1]

print(f"Correlation coefficient: {coef:.2f}")
```