مكتب

Lab Activity

Week 12: Debugging Workshop

Duration: 70 minutes — Work in pairs!

Name: ____________________ **Partner:** ____________________

Lab Goals

By the end of this lab, you will be able to:

- Quickly identify syntax, runtime, and logic errors
- Read error messages to locate problems fast
- Use print statements to debug systematically
- Work with a partner to fix buggy programs
- Apply debugging strategies to solve problems

Getting Started

1. Extract the given `week10_lab.zip` file to your desktop. This folder contains all the files you need for this lab.

1 Exercise 1: Let's Spot Error Types (8 minutes)

Without running the code first, for each of the following snoppets, find the type of error it has. Is it a syntax error, runtime error, or logic error?

Exercise 1:

Example 1: This will work fine

```python
name = "Ahmed"
print(f"Hello, {name}!")
```

Example 2: Syntax Error - Python can't understand

```python
print("Testing syntax error")
if True
    print("Missing something?")
```

Example 3: Runtime Error - Crashes while running

```python
number = 50
result = number / 0
print(f"Result: {result}")
```

Example 4: Logic Error - Runs but wrong result

```python
total_cost = 100
discount_percent = 10
final_price = total_cost - discount_percent
print(f"Final price: Rs.{final_price}")
```

For each example create a new file in Thonny and copy the code. Then run and fix all issues.

Checkpoint: Can you identify what's wrong with all examples? Show your teacher!

2 Exercise 2: Error Message Detective (10 minutes)

Let's practice reading error messages *(without running the code)*! For each snippet below, predict the error, then type and run to check:

Detective Case 1:

What error will this cause?

```python
greeting = "Hello
print(greeting)
```

Your prediction: ______________________________

Actual error type: ______________________________

Detective Case 2:

What about this one?

```python
age = input("Enter your age: ")  # User types 13
next_year = age + 1
print(f"Next year you'll be {next_year}")
```

Your prediction: ______________________________

Actual error type: ______________________________

Detective Case 3:

And this?

```python
for i in range(5):
print(i)
```

Your prediction: ______________________________

Line number with error: _______________

Tip: Always look at the line number in the error message first!

3 Exercise 3: Print Debug Practice (10 minutes)

Open the file `week12_ex3.py` in Thonny. This program is supposed to count total rupees from coins, but it has a bug. Let's use print statements to find where this coin counter goes wrong:

```
1  # Exercise 3: Debug the Coin Counter
2  print("=== Coin Counter ===")
3  
4  # This program should count total rupees from coins
5  ten_rupee_coins = int(input("How many 10 rupee coins? "))
6  five_rupee_coins = int(input("How many 5 rupee coins? "))
7  one_rupee_coins = int(input("How many 1 rupee coins? "))
8  
9  # Calculate total (has bugs!)
10 total = 0
11 total = ten_rupee_coins
12 total = five_rupee_coins
13 total = one_rupee_coins
14 
15 print(f"Total rupees: {total}")
```

Test with:

- Test with: 2 ten-rupee, 3 five-rupee, 4 one-rupee coins
- Should be: 20 + 15 + 4 = 39 rupees
- But shows: 4 rupees (only counts the last coins!)

Add print statements after each calculation to see where the total goes wrong:

- After line 11: Print what total equals
- After line 12: Print what total equals now
- After line 13: Print what total equals finally

✓ **Checkpoint:** Found the bug using prints? Fix it and show the working version!

Pair Programming: Switch who types every 5 minutes!

4 Exercise 4: Bug Hunt Challenge (12 minutes)

Open the file `week12_ex4.py`. This temperature converter has multiple bugs. Your task is to find and fix all of them!

```
print("=== Temperature Converter ==")
print("Convert between Celsius and Fahrenheit")

# Get user choice
choice = input("Enter C for Celsius to Fahrenheit, F for opposite: ")

if choice == "C"
    # Convert Celsius to Fahrenheit
    temp = input("Enter temperature in Celsius: ")

```

```
11     fahrenheit = (temp - 32) * 5/9
12     print(f"{temp} F = {fahrenheit} C")
13
14 elif choice = "F":
15     # Convert Fahrenheit to Celsius
16     temp = int(input("Enter temperature in Fahrenheit: "))
17
18     celsius = temp * 9 // 5 + 32
19     print(f"{temp} F = {celsius} C")
20
21 else:
22     print("Invalid choice! Enter C or F")
23
24 # Check if it's freezing
25 if temp < 0:
26     print("That's below freezing!")
```

Bug Tracking Table:

Bug #	Location	How to Fix
1	Line 7	
2		
3		
4		
5		
6		
7		
8		

Test your fixed program with:

- Choice: C, Temperature: 0 (should show 32°F)
- Choice: F, Temperature: 212 (should show 100°C)
- Choice: c (lowercase - should it work?)

✓ **Checkpoint:** All bugs fixed and program working? Excellent teamwork!

5 Exercise 5: Debug the Cricket Score Calculator (10 minutes)

Open the file week12_ex5.py and read the comments. This cricket score calculator has both syntax and logic errors. Your task is to find and fix them all!

```
# Exercise 5: Debug the Cricket Score Calculator
print("=== Cricket Score Calculator ===")
print("Calculate team's run rate and total score")

# Get match details (Has multiple bugs!)
overs_played = input("How many overs played? ")
runs_scored = int(input("Total runs scored: "))

# Calculate run rate (runs per over)
run_rate = runs_scored / overs_played
print(f"Current run rate: {run_rate} runs per over")

```

```python
# Predict final score (50 overs total)
overs_left = 50 - overs_played
predicted_runs = run_rate * overs_left
final_score = runs_scored + predicted_runs

# Display predictions
print(f"\nMatch Statistics:")
print(f"Overs played: {overs_played}/50")
print(f"Current score: {runs_scored}")
print(f"Run rate: {run_rate}")

# Check if good run rate (Bug: wrong comparison)
if run_rate => 6.0:
    print("Excellent run rate!")
elif run_rate >= 5.0:
    print("Good run rate!")
elif run_rate >= 4.0
    print("Average run rate")
else:
    print("Need to score faster!")

# Final score prediction
print(f"\nPredicted final score: {final_score}")

# Bonus calculation with bug
if overs_played < 20:
    print("Still in first half of innings")
    # Calculate required run rate for target of 300
    runs_needed = 300 - runs_scored
    overs_left = 50 - overs_played
    required_rate = runs_needed * overs_left
    print(f"Need {required_rate} runs per over for 300")
```

Bug Hunt Checklist:

- □ Type conversion missing
- □ What happens when you divide by a string?
- □ Syntax error in comparison operator
- □ Missing punctuation
- □ Wrong calculation(s) or formula(s) used

Test with: 10 overs, 65 runs (should show 6.5 run rate, predict 325 final score)

✓ **Checkpoint:** All cricket bugs fixed? Program calculates correctly? Show your teacher!

Pair Programming: Switch who types every 5 minutes!

6 Exercise 6: Debugging Race - Pizza Party Planner (15 minutes)

The Challenge: Your class is planning a pizza party! Open the file week12_ex6.py and read the comments. This program calculates costs but has some bugs. Race to fix them all!

```python
# Exercise 6: Pizza Party Cost Calculator - Full of Bugs!
print("=== Pizza Party Planner ===")
print("Calculate how many pizzas and total cost")

# Get party details
students = input("How many students in class? ")
teachers = int(input("How many teachers? "))

# Each person eats 2 slices, each pizza has 8 slices
total_people = students + teachers
slices_needed = total_people * 3

# Calculate pizzas needed
slices_per_pizza = 8
pizzas_needed = slices_needed / slices_per_pizza

# Should round up (e.g., 4.1 pizzas means we need 5)
if pizzas_needed > int(pizzas_needed):
    pizzas_needed = int(pizzas_needed) + 1
else
    pizzas_needed = int(pizzas_needed)

print(f"\nParty Details:")
print(f"Total people: {total_people}")
print(f"Slices needed: {slices_needed}")
print(f"Pizzas needed: {pizzas_needed}")

# Calculate cost
pizza_price = 800  # Rs.800 per pizza
drinks_price = 50  # Rs.50 per person

pizza_cost = pizzas_needed * pizza_price
drinks_cost = total_people + drinks_price
total_cost = pizza_cost + drinks_cost

# Apply discount for large orders
if pizzas_needed >= 5
    discount = 10  # 10% discount
    discount_amount = pizza_cost * discount / 100
    final_cost = total_cost - discount_amount
    print(f"\nLarge order discount: {discount}%")
    print(f"You save: Rs.{discount_amount}")
elif pizzas_needed >= 3:
    discount = 5
    discount_amount = total_cost * discount / 100
    final_cost = total_cost - discount_amount
    print(f"\nSmall discount: {discount}%")
    print(f"You save: Rs.{discount_amount}")
else:
    final_cost = total_cost
    print("\nNo discount (order 3+ pizzas for discount)")

# Cost per person
cost_per_person = final_cost / (total_people - 1)

# Display final costs
print("\n" + "-"*30)
print("COST BREAKDOWN")
print("-"*30)
print(f"Pizza cost: Rs.{pizza_cost}")
print(f"Drinks cost: Rs.{drinks_cost}")
print(f"Subtotal: Rs.{total_cost}")
print(f"Final total: Rs.{final_cost}")
```

```
64 print(f"Per person: Rs.{cost_per_person}")
65 print("-"*30)
66
67 # Check if within budget
68 budget = 15000
69 if final_cost < budget:
70     print(f"\n Within budget! Rs.{budget - final_cost} under budget")
71 elif final_cost = budget:
72     print("\n Exactly on budget!")
73 else:
74     print(f"\n Over budget by Rs.{final_cost + budget}")
```

Bug Tracking Table:

Bug #	Line	What's Wrong?
1	6	*Missing* `int()` *conversion for students input*
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		

Test with: 25 students, 2 teachers (should need 7 pizzas, get 10% discount)

Correct Output (all bugs fixed):

```
=== Pizza Party Planner ===
Calculate how many pizzas and total cost
How many students in class? 25
How many teachers? 2

Party Details:
Total people: 27
Slices needed: 54
Pizzas needed: 7

Large order discount: 10%
You save: Rs.695.0

------------------------------
COST BREAKDOWN
------------------------------
Pizza cost: Rs.5600
Drinks cost: Rs.1350
Subtotal: Rs.6950
Final total: Rs.6255.0
Per person: Rs.231.67
------------------------------

Within budget! Rs.8745.0 under budget
```

✓ **Checkpoint:** All bugs fixed? How long did it take? ____ minutes

7 Lab Summary

✓ **Checkpoint:** Final checkpoint - make sure you have:

- □ Identified all three types of errors
- □ Used print debugging to find bugs
- □ Fixed programs with multiple errors
- □ Worked successfully with your partner
- □ Shown all checkpoints to your teacher

Reflection (2 minutes)

Rate your debugging skills after today's practice:

Skill	😐 Need Help	🙁 Getting It	🙂 Got It!
Identifying error types	○	○	○
Reading error messages	○	○	○
Using print debugging	○	○	○
Fixing syntax errors	○	○	○
Finding logic errors	○	○	○

My best debugging tip from today: ____

The hardest bug I fixed was: ____

🙂 You're now a debugging detective! Great work!