Project 1: Simple Calculator with Modules

IT fundamentals: Python Programming Course

Assigned after Lecture 3

Due: 1 Week | Weight: 30% | Estimated Time: 4-6 hours

1 Project Overview

1.1 Main Purpose

Create a simple calculator application that demonstrates basic Python concepts: functions, modules, and file handling. This is a focused project designed to reinforce fundamental skills without overwhelming complexity.

1.2 Learning Objectives

By completing this project, you will:

- Apply functions to organize code
- Create and import custom modules
- Handle user input and basic error cases
- Save and load data using JSON files
- Practice writing clean, documented code

2 Requirements (Keep It Simple!)

2.1 Core Features (Required)

Your calculator must support:

- **Basic Operations**: Add, subtract, multiply, divide
- **Menu System**: Simple text-based menu
- **History**: Save last 5 calculations to a file
- **Input Validation**: Handle non-numeric inputs
- **Modular Code**: Split into 2-3 Python files

2.2 Project Structure (Simplified)

```
simple_calculator/
├── main.py              # Main program
├── calculator.py        # Calculator functions
├── history.py           # History management
└── calculations.json    # Saved calculations (created by program)
```

That's it! No complex directory structures or advanced features.

3 Skills Applied

3.1 From Lectures 1-2: Functions & File Handling

- Create functions for each math operation
- Use try/except for error handling
- Read/write JSON files for history

3.2 From Lecture 3: Modules

- Split code into separate .py files
- Import your own modules
- Use built-in modules (json, math if needed)

4 Starter Code (Copy and Build On This)

4.1 main.py - Start Here

```python
"""
Simple Calculator - Main Program
Fill in the missing parts!
"""
from calculator import add, subtract, multiply, divide
from history import save_calculation, load_history, show_history

def show_menu():
    print("\n=== Simple Calculator ===")
    print("1. Add")
    print("2. Subtract")
    print("3. Multiply")
    print("4. Divide")
    print("5. Show History")
    print("6. Quit")

def get_numbers():
    """Get two numbers from user with error handling"""
    try:
        a = float(input("Enter first number: "))
        b = float(input("Enter second number: "))
        return a, b
    except ValueError:
        print("Please enter valid numbers!")
        return None, None

def main():
    while True:
        show_menu()
        choice = input("Choose (1-6): ")

        if choice == '6':
            print("Goodbye!")
            break
        elif choice == '5':
            show_history()
        elif choice in ['1', '2', '3', '4']:
            a, b = get_numbers()
            if a is not None and b is not None:
                # TODO: Call the right function based on choice
                # TODO: Save the calculation to history
                pass
        else:
            print("Invalid choice!")

if __name__ == "__main__":
    main()
```

4.2 calculator.py - Math Functions

```python
"""
Calculator functions
Complete these functions!
"""

def add(a, b):
    """Add two numbers"""
    # TODO: Return a + b
    pass

def subtract(a, b):
    """Subtract b from a"""
    # TODO: Return a - b
    pass

def multiply(a, b):
    """Multiply two numbers"""
    # TODO: Return a * b
    pass

def divide(a, b):
    """Divide a by b, handle division by zero"""
    if b == 0:
        return "Error: Cannot divide by zero"
    # TODO: Return a / b
    pass

# Test your functions here
if __name__ == "__main__":
    print("Testing calculator functions...")
    print(f"2 + 3 = {add(2, 3)}")
    print(f"10 / 2 = {divide(10, 2)}")
    print(f"10 / 0 = {divide(10, 0)}")
```

4.3 history.py - Simple History Management

```python
"""
History management - keep it simple!
"""
import json
import os

HISTORY_FILE = "calculations.json"

def load_history():
    """Load history from file, return empty list if file doesn't exist"""
    try:
        if os.path.exists(HISTORY_FILE):
            with open(HISTORY_FILE, 'r') as file:
                return json.load(file)
        return []
    except:
        print("Could not load history file.")
        return []

def save_calculation(operation, result):
    """Save a calculation to history (keep only last 5)"""
    history = load_history()

    # Create new entry
    entry = {
        "operation": operation,
        "result": result
    }

    # Add to history
    history.append(entry)

    # Keep only last 5 calculations
    if len(history) > 5:
        history = history[-5:]

    # Save to file
    try:
        with open(HISTORY_FILE, 'w') as file:
            json.dump(history, file, indent=2)
    except:
        print("Could not save to history file.")

def show_history():
    """Display calculation history"""
    history = load_history()

    if not history:
        print("No calculations in history.")
        return

    print("\n=== Recent Calculations ===")
    for i, entry in enumerate(history, 1):
        print(f"{i}. {entry['operation']} = {entry['result']}")
```

5 What You Need to Complete

5.1 Step-by-Step Instructions

1. **Copy the starter code** into three separate files
2. **Complete the calculator functions** - just return the math results
3. **Connect main.py** - call the right function based on user choice
4. **Add history saving** - call save_calculation() after each operation
5. **Test everything** - make sure it works!
6. **Add comments** - explain what your code does

5.2 Example of Completed Code (for main.py)

```python
# In main.py, replace the TODO section:
if choice == '1':
    result = add(a, b)
    operation = f"{a} + {b}"
    save_calculation(operation, result)
    print(f"Result: {result}")
elif choice == '2':
    result = subtract(a, b)
    operation = f"{a} - {b}"
    save_calculation(operation, result)
    print(f"Result: {result}")
# ... and so on for multiply and divide
```

6 Testing Your Calculator

Test with these examples:

- Try: 10 + 5 = 15
- Try: 20 / 4 = 5
- Try: 10 / 0 = Error message
- Try: ”abc” as input = Error message
- Check: History shows last 5 calculations
- Check: History persists when you restart the program

7 Submission Requirements

7.1 What to Submit

- Your three Python files (main.py, calculator.py, history.py)
- A screenshot showing your calculator working
- One paragraph explaining what you learned

7.2 Grading Criteria (Simple!)

- **Works correctly (60%)**: Calculator performs basic operations
- **Uses modules (20%)**: Code is split into separate files with imports
- **Handles errors (10%)**: Doesn’t crash on bad input
- **History feature (10%)**: Saves and shows recent calculations

8 Tips for Success

- **Start small**: Get basic math working first
- **Test as you go**: Run your code frequently
- **Don’t overthink**: This should be simple!
- **Ask for help**: If stuck for more than 30 minutes
- **Copy the starter code exactly**: Don’t change the structure

9 Optional Extras (Only if you finish early)

If you complete the basic requirements and want to do more:

- Add square root using math.sqrt()
- Add a ”clear history” option
- Make the display prettier with formatting

Remember: This project should take 4-6 hours total. If it’s taking much longer, you’re probably overthinking it. Keep it simple and focus on the core requirements!