Simple Calculator Project - Week 1

Project Overview

Build a command-line calculator that performs basic arithmetic operations. This project will reinforce Python fundamentals and give you your first real portfolio piece.

Learning Objectives

- Practice functions and user input
- Handle errors gracefully
- Use control structures (if/else, while loops)
- Apply mathematical operations
- Create a user-friendly interface

Project Structure

```
coding-journey/
├── hello.py (already created)
├── calculator.py (new file)
└── README.md (we'll create this)
```

Step-by-Step Implementation

Step 1: Create the Basic Calculator (15 minutes)

Create a new file called `calculator.py` in your coding-journey folder:

python

```python
def add(x, y):
    """Add two numbers"""
    return x + y

def subtract(x, y):
    """Subtract two numbers"""
    return x - y

def multiply(x, y):
    """Multiply two numbers"""
    return x * y

def divide(x, y):
    """Divide two numbers"""
    if y == 0:
        return "Error: Cannot divide by zero!"
    return x / y

def calculator():
    """Main calculator function"""
    print("🔢 Simple Calculator")
    print("==================")

    while True:
        # Display menu
        print("\nSelect operation:")
        print("1. Add")
        print("2. Subtract")
        print("3. Multiply")
        print("4. Divide")
        print("5. Exit")

        choice = input("\nEnter choice (1-5): ")

        if choice == '5':
            print("Thanks for using the calculator! 👋 ")
            break

        if choice in ['1', '2', '3', '4']:
            try:
                num1 = float(input("Enter first number: "))
                num2 = float(input("Enter second number: "))
```

```
            if choice == '1':
                result = add(num1, num2)
                print(f"\n{num1} + {num2} = {result}")

            elif choice == '2':
                result = subtract(num1, num2)
                print(f"\n{num1} - {num2} = {result}")

            elif choice == '3':
                result = multiply(num1, num2)
                print(f"\n{num1} × {num2} = {result}")

            elif choice == '4':
                result = divide(num1, num2)
                print(f"\n{num1} ÷ {num2} = {result}")

        except ValueError:
            print("❌ Error: Please enter valid numbers!")

    else:
        print("❌ Invalid choice! Please select 1-5.")

# Run the calculator
if __name__ == "__main__":
    calculator()
```

Step 2: Test Your Calculator (10 minutes)

1. **Save the file** (`calculator.py`)
2. **Run it in Terminal:**

   bash

   ```
   python3 calculator.py
   ```

3. **Test each operation:**
   - Try addition: 5 + 3
   - Try subtraction: 10 - 4
   - Try multiplication: 6 × 7
   - Try division: 15 ÷ 3
   - Try division by zero: 10 ÷ 0
   - Try invalid input: letters instead of numbers

- Exit with option 5

Step 3: Add Advanced Features (20 minutes)

Let's enhance your calculator with more features. Add these functions to your `calculator.py`:

```python
import math

def power(x, y):
    """Calculate x to the power of y"""
    return x ** y

def square_root(x):
    """Calculate square root"""
    if x < 0:
        return "Error: Cannot calculate square root of negative number!"
    return math.sqrt(x)

def percentage(x, y):
    """Calculate x% of y"""
    return (x / 100) * y
```

Update your menu in the `calculator()` function:

```python
    print("\nSelect operation:")
    print("1. Add")
    print("2. Subtract")
    print("3. Multiply")
    print("4. Divide")
    print("5. Power (x^y)")
    print("6. Square Root")
    print("7. Percentage (x% of y)")
    print("8. Exit")
```

And add the new cases:

python

```python
    if choice == '8':
        print("Thanks for using the calculator! 👋 ")
        break

    if choice in ['1', '2', '3', '4', '5', '7']:
        # ... existing code for two numbers ...

    elif choice == '6':
        try:
            num = float(input("Enter number: "))
            result = square_root(num)
            print(f"\n√{num} = {result}")
        except ValueError:
            print("❌ Error: Please enter a valid number!")
```

Step 4: Create a README File (10 minutes)

Create a `README.md` file in your coding-journey folder:

markdown

```markdown
# My Coding Journey

Welcome to my coding journey! This repository tracks my progress as I learn software engineering and AI automat

## Projects

### 1. Simple Calculator
A command-line calculator that performs basic arithmetic operations.

**Features:**
- Addition, subtraction, multiplication, division
- Power calculations (x^y)
- Square root calculations
- Percentage calculations
- Error handling for invalid inputs
- User-friendly interface

**How to run:**
```bash
python3 calculator.py
```
```

Skills learned:

- Python functions
- User input handling
- Error handling with try/except
- While loops and control structures
- Mathematical operations

Learning Plan

Following an 8-month comprehensive plan to become job-ready in software engineering and AI automation.

Current Progress: Week 1 - Python Fundamentals ✅

Step 5: Push to GitHub (5 minutes)

```bash
git add .
git commit -m "Add simple calculator with advanced features and README"
git push
```

Challenge Extensions (Optional)

If you finish early, try these challenges:

Challenge 1: Memory Feature

Add the ability to store and recall previous results:

```python
def calculator():
    memory = 0

    # Add these menu options:
    print("9. Memory Store (MS)")
    print("10. Memory Recall (MR)")
    print("11. Memory Clear (MC)")
```

Challenge 2: History Feature

Keep track of the last 5 calculations and display them.

Challenge 3: Scientific Calculator

Add trigonometric functions (sin, cos, tan) and logarithms.

What You've Learned

By completing this project, you've practiced:

- ✅ Python functions and parameters
- ✅ User input and data validation
- ✅ Error handling with try/except
- ✅ While loops and conditional statements
- ✅ Mathematical operations and the math module
- ✅ Code organization and documentation
- ✅ Git version control
- ✅ Creating professional README files

Next Steps

Once you complete this calculator:

1. Test all features thoroughly
2. Push your code to GitHub
3. Share your project (post about it on LinkedIn!)
4. Move on to the number guessing game
5. Start planning your expense tracker project

Time to complete: 1-2 hours **Difficulty:** Beginner **Skills:** Python fundamentals, functions, error handling