**Python Programming Worksheet**

**1. User Input and Output**

**Task**: Create a program that asks the user for their name and then greets them with a personalized message.

**Instructions**: Write a program that prompts the user to enter their name. After the user enters their name, display a message that includes their name in a friendly greeting.

---

**2. Working with Variables**

**Task**: Assign values to variables and perform basic arithmetic.

**Instructions**: Create variables to represent different pieces of data. Then, use those variables to perform simple arithmetic operations like addition, subtraction, multiplication, and division. Finally, display the results.

---

**3. Comments in Code**

**Task**: Document your code with comments.

**Instructions**: Write a program that includes at least three lines of code and add comments to explain what each part of the code does. Comments should describe the purpose of the code and any important details.

---

**4. String Operations**

**Task**: Combine multiple strings into a single message.

**Instructions**: Write a program that takes two or more pieces of text and joins them together to form a complete sentence. Display the combined text to the user.

---

**5. Type Conversion**

**Task**: Convert user input into different data types.

**Instructions**: Create a program that asks the user for a number. Then, convert that input into a different type (e.g., from a string to an integer) and perform an operation with it. Show the user the result of that operation.

---

**6. Arithmetic Operations with Real Numbers**

**Task**: Perform calculations with decimal numbers.

**Instructions**: Write a program that calculates the price of an item after adding sales tax. Ask the user for the price of the item, calculate the total cost after adding a percentage for tax, and display the final amount.

---

**7. Understanding Order of Operations**

**Task**: Calculate expressions with multiple operations.

**Instructions**: Write a program that performs a calculation involving addition, multiplication, and parentheses. Explore how changing the order of operations (using parentheses) affects the result.

---

**8. Documenting Code**

**Task**: Explain your program with comments.

**Instructions**: Write a program that performs a simple task and includes comments to describe what each section of the program does. Your comments should explain the overall purpose of the program as well as individual lines of code.

---

**9. Creating Functions**

**Task**: Define a function that performs a calculation.

**Instructions**: Create a function that takes two numbers as input and returns the result of a mathematical operation (e.g., multiplication or division). Call this function from your main program and display the result.

---

**10. Using Constants in Calculations**

**Task**: Calculate an area using a constant.

**Instructions**: Write a program that calculates the area of a circle. Define a constant for the value of pi, ask the user for the radius of the circle, and use the formula for the area of a circle to calculate and display the result.

---

**11. Combining Arithmetic and Variables**

**Task**: Use variables to store data and perform calculations.

**Instructions**: Create a program that asks the user for two measurements (like length and width) and then calculates an area or perimeter. Display the result to the user.

---

**12. Building Dynamic Text**

**Task**: Create a program that generates personalized messages.

**Instructions**: Write a program that combines user input with preset text to create a personalized message. For example, generate a welcome message that includes the user's name or favorite color.

---

**13. Gathering Input from Users**

**Task**: Create a program that interacts with the user.

**Instructions**: Design a program that asks the user a series of questions, gathers their responses, and then displays those responses in a summary format.

---

**14. Creating and Using Functions**

**Task**: Break down your program into smaller tasks using functions.

**Instructions**: Write a program that performs a task by calling a function. Ensure the function returns a value that is used elsewhere in the program.

---

**15. Introduction to Turtle Graphics**

**Task**: Draw basic shapes using turtle graphics.

**Instructions**: Use a turtle graphics library to draw simple shapes like squares, triangles, or lines. Control the turtle’s movement to create patterns or designs on the screen.

---

**16. Drawing Complex Shapes**

**Task**: Create a design using loops and turtle graphics.

**Instructions**: Write a program that uses loops to repeat drawing commands with the turtle graphics library. Create a complex shape, such as a star or a polygon, by repeating simple movements.

---

**17. Customizing Turtle Graphics**

**Task**: Change the appearance of your turtle drawings.

**Instructions**: Modify your turtle program to use different colors for the pen. Experiment with lifting the pen to create designs with unconnected lines.

---

**18. Moving the Turtle Without Drawing**

**Task**: Move the turtle to different locations without drawing.

**Instructions**: Write a program that uses turtle graphics to move the turtle to different parts of the screen without drawing. Then, have the turtle draw only in specific areas.

**Project: Virtual Greeting Card with Dynamic Design and Interaction**

**Project Overview:**

Create a Python program that generates a virtual greeting card. The card will include a personalized message, basic calculations, and a simple design drawn using turtle graphics. The program will prompt the user for input, perform arithmetic operations, and use functions to organize the code. The project should incorporate every concept from the chapter at least once.

---

**Step-by-Step Instructions:**

**1. Greeting and Personalization**

**Task**: Start the program by asking the user for their name and a special occasion (e.g., birthday, anniversary).

**Implementation**:

Use input() to gather the user's name and the occasion.

Store these inputs in variables.

Create a personalized greeting message using string concatenation and display it using print().

**2. Basic Calculations**

**Task**: Ask the user for a number (e.g., their age or the number of years they've celebrated the occasion). Use this number in a simple arithmetic operation.

**Implementation**:

Prompt the user to enter a number using input() and convert it to an integer using int().

Perform a calculation using this number (e.g., multiply by 2, add 10, etc.).

Display the result with a message explaining the calculation.

**3. Documenting the Code**

**Task**: Write comments throughout the program to explain each part of the code.

**Implementation**:

Use comments to describe the purpose of each section, what each function does, and what specific lines of code are doing.

Ensure that each major block of code is well-documented.

**4. Function Creation**

**Task**: Create functions to organize the code. For example, one function could handle the greeting and another could perform calculations.

**Implementation**:

Define at least two functions: one for displaying the greeting message and another for performing the arithmetic operations.

Call these functions from the main part of the program.

**5. Constants and Variables**

**Task**: Define a constant for a value that doesn't change (e.g., the multiplier or adder used in your calculation).

**Implementation**:

Use an uppercase variable name for your constant.

Use this constant in your arithmetic operations.

**6. String Operations**

**Task**: Create a dynamic message that incorporates the user's input.

**Implementation**:

Combine the user's name and occasion with additional text to create a complete greeting.

Use string concatenation to build the message.

**7. Turtle Graphics Design**

**Task**: Use turtle graphics to draw a simple design on the virtual card.

**Implementation**:

Create a turtle object.

Draw shapes like a rectangle (for the card's border), a circle (decorative element), or a star.

Use loops to repeat drawing commands and create patterns.

Experiment with changing pen colors and moving the turtle without drawing.

**8. Order of Operations**

**Task**: Perform a calculation that demonstrates the order of operations.

**Implementation**:

Use a mix of addition, multiplication, and parentheses in a calculation related to the user's input.

Display the result, explaining how the order of operations affected the calculation.

**9. Pen Control in Turtle Graphics**

**Task**: Draw parts of the design without connecting the lines.

**Implementation**:

Use penup() and pendown() to lift and lower the turtle's pen.

Move the turtle to different positions on the screen to create separate elements of the design.

**10. Final Display**

**Task**: Ensure that all elements are displayed neatly and clearly.

**Implementation**:

Use print() to display the final greeting message.

Ensure the turtle graphics window stays open until the user closes it.

---

**Expected Output:**

A virtual greeting card that:

Greets the user by name and mentions the special occasion.

Displays the result of a calculation related to the user's input.

Draws a simple but attractive design using turtle graphics, possibly with different shapes and colors.

Demonstrates basic arithmetic, variable assignment, functions, string manipulation, and turtle graphics.

---

**Example Walkthrough:**

1. **Input**:
    - Name: "Alice"

- Occasion: "Birthday"
- Age: 25

2. **Greeting Message**:
    - "Happy Birthday, Alice!"
3. **Calculation**:
    - "Next year, you'll be 26 years old!"
4. **Turtle Graphics**:
    - The screen shows a rectangular border around the virtual card, with a star or circle drawn inside as decoration.

---

This project encompasses all the major concepts from the chapter and challenges the student to integrate them into a cohesive program. Each step is designed to reinforce learning while allowing for creativity and personalization.