Class 03

Class Plan

1. Announcements: Canvas page, email: fshahsavari@sdsu.edu
2. Homework Solution
3. Classroom Circular set up (groups of 5-6)
4. Review Conditions and Iterations

5. Functions:
   - Global vs. Local variables
   - Default arguments
   - Documentation strings

Homework solution (**Notebook**)

- **"Write a program that asks the user to input a number. Then, the program t ells if the number is odd or even."**

Functions

- Function is a block of code that performs a specific task.
- Functions make your code more modular and readable.

Function creation

Example:

```
def greet(name):
        return "Hello, " + name
```

Calling a Function

Practice:

```
def greet(name):
        return "Hello, " + name
```

Example: greet("Alice")

Global Variables

- Global Variables:

- Global variables are defined outside of any function or block, at the top level of the program.
- They can be accessed and modified from anywhere within the program, including inside functions.
- Global variables are created when they are first assigned a value and exist until the program terminates.

Local Variables

- Local Variables:

- Local variables are defined inside a function or a block.
- They are accessible only within the function or block where they are defined.
- Local variables are created when the function or block is executed and destroyed when the function or block execution ends.

Local vs. Global variables

```
x=20
def my_function():
    x = 10  # Local variable
    print(x)

my_function()  # Output: 10
```

Default Arguments

- Defining a Function with Default Arguments:

```
def greet(name, message="Hello"):
    print(message, name)
```

In this example, the greet() function has two parameters: name and message. The message parameter has a default value of "Hello". If no value is provided for message when calling the function, it will default to "Hello".

Calling a Function with a default argument

Practice:

```
def greet(name, message="Hello"):
        print(message, name)
```

Example:

greet(“John”, message)

greet(“John”, “How are you doing?”)

Documentation Strings

- In Python, a documentation string, also known as a docstring, is a string literal specified as the first statement within a function, module, class, or method definition.

- Docstrings serve as a form of documentation, providing information about the purpose, usage, and behavior of the function.

Documentation Strings

```
def calculate_sum(a, b):
    """
    Calculates the sum of two numbers.

    Parameters:
    a (int): The first number.
    b (int): The second number.

    Returns:
    int: The sum of the two numbers.
    """
    return a + b
```

Documentation Strings

Using Good Practices for Docstrings:

- Use the triple quotes (""") to define multiline docstrings. This allows for detailed explanations and examples.
- Include a summary of the function's purpose in a single line.
- Describe the function's parameters, specifying their types and any constraints or requirements.
- Explain the return value and its type, if applicable.
- If the function has any side effects or raises exceptions, document those as well.

Notebook Time!

Next Class

- Debugging with Pycharm
- Object-Oriented Programming