3. Worksheet: Introduction to Variables, Parameters, and Parameter Passing

Objective:

Understand the concepts of global and local variables, formal and actual parameters, and parameter passing by value and by reference in Python.

---

1. Definitions:

- **Global Variable:** A variable declared outside of a function. It is accessible from any function in the code.
- **Local Variable:** A variable declared inside a function. It can only be accessed within that function.
- **Formal Parameter:** The variables listed in the function definition. They act as placeholders for the values passed into the function.
- **Actual Parameter:** The values passed into the function when it is called. They are used to initialize the formal parameters.
- **Pass by Value:** A method of parameter passing where a copy of the actual parameter's value is passed. Changes made to the parameter inside the function do not affect the original value.
- **Pass by Reference:** A method of parameter passing where a reference to the actual parameter's memory location is passed. Changes made to the parameter inside the function affect the original value.

---

2. Practical Example: Friend Finder App

See previous worksheet for a reminder of the code

---

3. Parameter Concepts:

a. Formal and Actual Parameters

```
# Function with formal parameters
def greet(name, age):
    print(f"Hello, {name}! You are {age} years old.")

# Calling the function with actual parameters
greet("Alice", 25)
```

Questions:

1. Identify the formal parameters in the function definition.
2. Identify the actual parameters in the function call.
3. What will be the output of the code?

---

b. Pass by Value

```
def modify_value(num):
    num = num * 2
    return num

value = 10
result = modify_value(value)
print(value, result)
```

Questions:

1. What is the value of the `value` variable after the function call?
2. What will be the output of the code?

---

c. Pass by Reference

```
def add_friend(friends_list, friend_name):
```

```
    friends_list.append(friend_name)

my_friends = ["Alice", "Bob"]
add_friend(my_friends, "Charlie")
print(my_friends)
```

Questions:

1. How does the `add_friend` function modify the `my_friends` list?
2. What will be the output of the code?

---

4. Exercises:

1. Complete the below function that takes a list of numbers and a number as parameters. The function should multiply each number in the list by the given number. Is this pass by value or pass by reference?

```
def multiply_list(list, multiplier_number):
  # This will iterate from 0 to the length of the list
  for count in range(len(list)):
    list[count] = # Multiply here...
  return list

list = [0,1,2]
print(multiply_list(list,10))
```

1. Implement a function that takes a string and returns a modified string without changing the original string. Is this pass by value or pass by reference?

---

5. Reflection:

1. Why is it important to understand the difference between pass by value and pass by reference?
2. In what scenarios would you prefer to use pass by value over pass by reference and vice versa?

Objective:

Understand the difference between global and local variables and their practical application in Python.

---

1. Definitions:

- **Global Variable:** A variable declared outside of a function. It is accessible from any function in the code.
- **Local Variable:** A variable declared inside a function. It can only be accessed within that function.

---

2. Practical Example: Friend Finder App

a. User Registration

```
registered_users = []

def register(username, password):
    user_data = {
        'username': username,
        'password': password
    }
    registered_users.append(user_data)

register("Alice", "password123")
print(registered_users)
```

Questions:

1. Identify the global variable in the above code.
2. Identify the local variables in the above code.
3. What will be the output of the code?

---

b. User Login

```
logged_in_user = None

def login(username, password):
    global logged_in_user
    for user in registered_users:
        if user['username'] == username and user['password'] ==
            logged_in_user = username
            return True
    return False

if login("Alice", "password123"):
    print(f"{logged_in_user} is now logged in!")
```

Questions:

1. Why did we use the `global` keyword in the `login` function?
2. What will be the output if the login is successful?
3. What will be the value of `logged_in_user` if the login fails?

---

c. Adding Friends

```
friend_requests = {}

def send_friend_request(sender, receiver):
    if receiver in friend_requests:
        friend_requests[receiver].append(sender)
    else:
        friend_requests[receiver] = [sender]

send_friend_request("Alice", "Bob")
print(friend_requests)
```

Questions:

1. What is the purpose of the `friend_requests` dictionary?
2. What will be the output after sending a friend request?

---

d. Accepting Friend Requests

```
friends_list = {}

def accept_friend_request(receiver, sender):
    if sender in friend_requests[receiver]:
        # Add to friends list
        if receiver in friends_list:
            friends_list[receiver].append(sender)
        else:
            friends_list[receiver] = [sender]

        if sender in friends_list:
            friends_list[sender].append(receiver)
        else:
            friends_list[sender] = [receiver]

        # Remove from friend requests
        friend_requests[receiver].remove(sender)

accept_friend_request("Bob", "Alice")
print(friends_list)
```

Questions:

1. How does the `accept_friend_request` function modify the global variables?
2. What will be the output after accepting a friend request?

---

3. Exercises:

1. Add a function to decline a friend request. How would you modify the global `friend_requests` dictionary?
2. Implement a function to display all friends of a user. Which global variable would you access?
3. Create a function to change a user's password. Consider the global `registered_users` list for this task.

---

4. Reflection:

1. Why is it important to differentiate between global and local variables?
2. In what scenarios would you prefer to use a local variable over a global variable?