Section 3: Python Functions, Scope and Errors

Contents

Section 1: Functions and Scopre

Exercise 1: Functions in Python

Task 1: Greeting message

Create a function that takes a list of names and outputs each name with a greeting.

```
friend_list = ["John", "Jane", "Jack"]

def greet_friends(friend_list):
    for name in friend_list:
        print(f"Hello {name}!")

greet_friends(friend_list)
```

Output:

```
Hello John!
Hello Jane!
Hello Jack!
```

Task 2: Tax Calculation

Create a function that calculates tax with a given income and tax rate. Try this with different incomes and tax rates.

Version 1:

```
income = 50000
tax_rate = 0.2

def calculate_tax(income, tax_rate):
    tax_result = income * tax_rate
    tax_result = round(tax_result, 2)
    print (f"The tax is £{tax_result}")

calculate_tax(income, tax_rate)
```

Output 1:

```
The tax is £10000.0
```

Version 2:

```
income = 70000
tax_rate = 0.4

def calculate_tax(income, tax_rate):
    tax_result = income * tax_rate
    tax_result = round(tax_result, 2)
    print (f"The tax is £{tax_result}")

calculate_tax(income, tax_rate)
```

Output 2:

```
The tax is £28000.0
```

Task 3: Compound Interest Calculator Function

Create a function that calculates the total amount of money earned by an investment each year.

```
def compound_interest(principal, duration, interest_rate):
    print(f"Initial investment is: £{principal}")
    if interest_rate < 0 or interest_rate > 1:
        print("Please enter a number between 0 and 1")
        return None

    if duration < 0:
        print("Please enter a positive number of years")
        return None

    for year in range(1, duration + 1):
         total_for_the_year = principal * (1 + interest_rate) ** year
         total_for_the_year = round(total_for_the_year, 2)
         print(f"The total for the year {year} is £{total_for_the_year}")

    return int(total_for_the_year)

compound_interest(1000,5,0.03)
```

Output:

```
Initial investment is: £1000
The total for the year 1 is £1030.0
The total for the year 2 is £1060.9
The total for the year 3 is £1092.73
The total for the year 4 is £1125.51
The total for the year 5 is £1159.27
1159
```

Section 2: Assertations and Errors

Exercise 3: Assertions

Task 1: assertion

Write a line of code in the format assertion where the expression is true. Write a second code line where the expression is false.

```
assert 1 + 1 == 2

assert 1 + 1 == 3
```

Output gives an Assertation Error

Exercise 4: Identifying and Fixing Common Errors

Task: Fixing Errors

Syntax Error:

Original code:

```
pritn("Hello, World!")
```

Fixed code:

```
print("Hello, World!")
```

Name Error:

Original code:

```
print("My favorite color is", favorite_color)
```

Fixed code:

```
favorite_color = "Blue"
print("My favorite color is", favorite_color)
```

Value Error:

Original code:

```
number1 = "5"
number2 = 3
result = number1 + number2
print("The sum is:", result)
```

Fixed code:

```
number1 = 5
number2 = 3
result = number1 + number2
print("The sum is:", result)
```

Index Error:

Original code:

```
fruits = ["apple", "banana", "cherry"]
print(fruits[3])
```

Fixed code:

```
fruits = ["apple", "banana", "cherry"]
print(fruits[1])
```

Indentation Error:

Original code:

```
time = 11
if time < 12
print("Good morning!")
```

```
time = 11
if time < 12:
    print("Good morning!")
```

Section 3: Your first larger-scale Python programme

Exercise 4: Complete Template Program

Task: To-Do list manager

Create a to-do list manager with the following functionalities:

- Initalise an empty list
- View current tasks in the lit
- Remove a task from the list
- Quit and exit the program

```
# Initialize an empty list to store tasks
tasks = []

# Function to add a task to the list
def add_task():
    new_task = input("Please input a new task: ")
    tasks.append(new_task)
```

```python
# Function to view current tasks in the list
def view_tasks():
    if tasks == []:
        print("You have no tasks")
        return

    for i, task in enumerate(tasks, start=1):
        print(f"{i}. {task}")

# Function to remove a task from the list
def remove_task():
    if tasks == []:
        print("No valid tasks to remove")
        return

    task_removed = input("Which task do you wish to remove? ")

    if task_removed not in tasks:
        print("Not a valid task")
        return

    tasks.remove(task_removed)

# Main program loop
while True:
    print("To-Do List Manager")
    print("1. Add a task")
    print("2. View tasks")
    print("3. Remove a task")
    print("4. Quit")

    choice = input("Enter your choice: ")

    if choice == "1":
        add_task()
        print("Your new task list is:")
        view_tasks()
    elif choice == "2":
        print("Your tasks are:")
        view_tasks()
    elif choice == "3":
        remove_task()
        print("Your tasks are:")
        view_tasks()
    elif choice == "4":
        break
    else:
        print("Invalid choice. Please try again.")
```

Output

```
To-Do List Manager
1. Add a task
2. View tasks
3. Remove a task
4. Quit
Enter your choice: 1
Please input a new task: Complete logbook
Your new task list is:
1. Complete logbook
To-Do List Manager
1. Add a task
2. View tasks
3. Remove a task
4. Quit
Enter your choice: 1
Please input a new task: Send logbook to group
Your new task list is:
1. Complete logbook
2. Send logbook to group
To-Do List Manager
1. Add a task
2. View tasks
3. Remove a task
4. Quit
Enter your choice: 2
Your tasks are:
1. Complete logbook
2. Send logbook to group
To-Do List Manager
1. Add a task
2. View tasks
3. Remove a task
4. Quit
Enter your choice: 3
Which task do you wish to remove? Complete logbook
Your tasks are:
1. Send logbook to group
To-Do List Manager
1. Add a task
2. View tasks
3. Remove a task
4. Quit
Enter your choice: 4
```