



School of Computer Science

COMP 2067 Programming for Beginners Summer 2023

Assignment 1

Copyright Notice

This assignment contains copyrighted material that is the property of the instructor. Any unauthorized distribution, reproduction or sharing of this material, including uploading to CourseHero, Chegg, StuDocu or other websites, is **strictly prohibited** and may be subject to legal action. Students are granted access to this material solely for their personal use and may not use it for any other purpose without the express written consent of the instructor. Thank you for respecting the intellectual property rights of the instructor.

This assignment is designed to consolidate your knowledge of Python programming and includes topics from Ch. 1 "Introduction to Python," Ch. 2 "Variables and Data Types," Ch. 3. "Operators and Expressions," Ch. 4. "Conditional Statements," Ch. 5 "Loops," and Ch. 6 "Functions."

I. Example

Gym Membership Calculator

Develop a program to calculate the membership fees for a gym based on various criteria. The program should include the following components:

1. Write a function called **calculate_membership_fee** that takes in the following parameters:
 - **age**: The age of the person (integer)
 - **membership_type**: The type of membership (string). The options are "Standard", "Premium", and "VIP".
 - **duration**: The duration of the membership in months (integer)
2. Inside the **calculate_membership_fee** function, implement the following logic:
 - If the person is below 18 years old, display a message saying that gym membership is not available for minors.
 - If the person is between 18 and 30 years old (inclusive), apply a discount of 10% on the membership fee.
 - If the membership type is "Standard", calculate the membership fee as \$50 per month.
 - If the membership type is "Premium", calculate the membership fee as \$75 per month.
 - If the membership type is "VIP", calculate the membership fee as \$100 per month.
 - Multiply the membership fee by the duration to get the total membership fee.
3. After calculating the membership fee, return the total fee from the **calculate_membership_fee** function.
4. Write a main program that prompts the user to enter their age, membership type, and duration of membership. Call the **calculate_membership_fee** function with the provided inputs and display the total fee to the user.
5. Test your program with different inputs to ensure it handles different scenarios correctly.

Source code:

```
def calculate_membership_fee(age, membership_type, duration):
    # Check if the age is less than 18
    if age < 18:
        print("Gym membership is not available for minors.")
        return

    # Initialize discount as 0
    discount = 0

    # Check if the age is between 18 and 30 (inclusive)
    if 18 <= age <= 30:
        discount = 0.1 # 10% discount for age between 18 and 30

    # Determine the membership fee based on the membership type
    if membership_type == "Standard":
        membership_fee = 50
    elif membership_type == "Premium":
        membership_fee = 75
    elif membership_type == "VIP":
        membership_fee = 100
    else:
        # If the membership type is invalid, print an error message and return from the function
        print("Invalid membership type.")
        return

    # Calculate the total fee by multiplying the membership fee with the duration
    total_fee = membership_fee * duration

    # Apply the discount to the total fee
    total_fee -= total_fee * discount

    # Return the total fee
    return total_fee

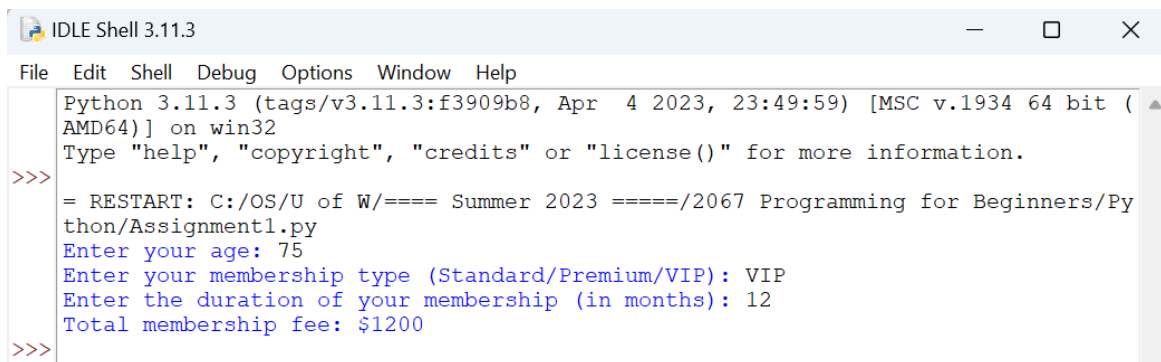
# Main program

# Prompt the user to enter their age, membership type, and duration
age = int(input("Enter your age: "))
membership_type = input("Enter your membership type (Standard/Premium/VIP): ")
duration = int(input("Enter the duration of your membership (in months): "))

# Call the calculate_membership_fee function with the provided inputs
total_fee = calculate_membership_fee(age, membership_type, duration)

# Check if the total_fee is not None (indicating a valid membership type)
if total_fee is not None:
    # Print the total membership fee
    print(f"Total membership fee: ${total_fee}")
```

Output:



```
IDLE Shell 3.11.3
File Edit Shell Debug Options Window Help
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> = RESTART: C:/OS/U of W/==== Summer 2023 =====/2067 Programming for Beginners/Python/Assignment1.py
Enter your age: 75
Enter your membership type (Standard/Premium/VIP): VIP
Enter the duration of your membership (in months): 12
Total membership fee: $1200
>>>
```

Fig. 1. Sample output

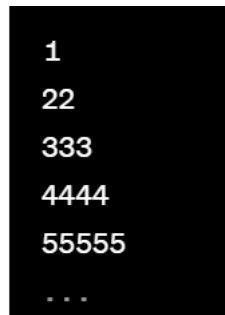
Explanation:

1. The **calculate_membership_fee** function takes in three parameters: **age**, **membership_type**, and **duration**. This function calculates and returns the total membership fee based on the given inputs.
2. The first conditional statement checks if the age is less than 18. If it is, it prints a message stating that gym membership is not available for minors and returns from the function.
3. The next conditional statement checks if the age is between 18 and 30 (inclusive). If it is, a discount of 10% (0.1) is applied.
4. The subsequent conditional statements determine the membership fee based on the membership type. If the membership type is invalid, an error message is printed and the function returns.
5. The total fee is calculated by multiplying the membership fee with the duration.
6. The discount is applied to the total fee by subtracting the discounted amount from the total fee.
7. Finally, the main program prompts the user to enter their age, membership type, and duration. It calls the **calculate_membership_fee** function with the provided inputs, and if the **total_fee** is not None (indicating a valid membership type), it prints the total membership fee.

II. Exercises

Exercise 1: Number Pattern

Write a function named `number_pattern` that takes a positive integer n as input and prints the following pattern:



```
1
22
333
4444
55555
...
```

Fig. 2. Sample output for Exercise 1

The number of rows in the pattern should be equal to n . Use loops and conditional statements to generate the pattern. Test your function with different values of n .

Exercise 2: Rock-Paper-Scissors Game

Write a function named `rock_paper_scissors` that allows a user to play the classic game of Rock-Paper-Scissors against the computer. The function should prompt the user for their choice (rock, paper, or scissors), generate a random choice for the computer, compare the choices, and determine the winner. Use loops, conditional statements, and functions to implement the game. Test your function by playing the game multiple times.

```

Enter your choice (rock, paper, or scissors): rock
Your choice: rock
Computer's choice: paper
Computer wins!
Do you want to play again? (yes/no): yes
Enter your choice (rock, paper, or scissors): paper
Your choice: paper
Computer's choice: paper
It's a tie!
Do you want to play again? (yes/no): yes
Enter your choice (rock, paper, or scissors): scissors
Your choice: scissors
Computer's choice: rock
Computer wins!
Do you want to play again? (yes/no): no

```

Fig. 3. Sample output for Exercise 2

Exercise 3: Banking Application

Write a program that simulates a simple banking application. The program should allow users to perform basic banking operations such as account balance inquiry, depositing funds, and withdrawing funds. The program should also keep track of multiple user accounts.

Requirements:

1. Implement a function to create a new bank account with an initial balance.
2. Implement a function to display the account balance for a given account number.
3. Implement a function to deposit funds into a specific account.
4. Implement a function to withdraw funds from a specific account, ensuring that the account has sufficient balance.
5. Use a loop to continuously prompt the user for operations until they choose to exit.

```

Welcome to the Banking Application!

1. Create a new account
2. Check account balance
3. Deposit funds
4. Withdraw funds
5. Exit

Enter your choice: 1

Enter initial balance: 1000
Account created successfully! Account number is 123456.

Enter your choice: 2

Enter account number: 123456
Account balance: $1000

Enter your choice: 3

Enter account number: 123456
Enter amount to deposit: 500
$500 deposited successfully. New balance: $1500

Enter your choice: 4

Enter account number: 123456
Enter amount to withdraw: 2000
Insufficient funds! Your current balance is $1500.

Enter your choice: 4

Enter account number: 123456
Enter amount to withdraw: 1000
$1000 withdrawn successfully. New balance: $500

Enter your choice: 5

Thank you for using the Banking Application!

```

Note: This program should contain only sample values. No real values should be provided for account numbers, balance, holder names, or any other sensitive information.

Fig. 4. Sample output for Exercise 3

III. Submission:

1. You will earn a maximum of 15% for submitting your report and source code within the specified deadline.
2. You must submit: (i) a report (in PDF or word), in which you provide **all the exercises**, source code and outputs with comments and explanations (see example, pp.1-3), (ii) all **three** Python files needed to run your programs (*.py). **Do not upload your source code to Brightspace as a single zip file.** Such submissions will not be accepted and **will result in an automatic grade of zero**.
3. Marks will be **deducted** if comments/explanations/outputs are missing.
4. Submissions after the deadline **will not be accepted**.
5. Unlimited resubmissions are allowed. But keep in mind that we will consider the last submission. That means that if you resubmit after the deadline, a penalty will be applied, even if you submitted an earlier version on time.

IV. Academic Integrity.

Plagiarism is a serious offense and can result in severe consequences such as receiving a grade of **zero** on the assignment/lab or even being asked to leave the program.

Copying or using someone else's code is considered plagiarism. This includes using code from online sources, previous labs/assignments, or from other students. Even if you have modified the code, our antiplagiarism software can still show it as plagiarism.

To avoid plagiarism, make sure to always use your own words and ideas when writing source code. Additionally, always check with your instructor to make sure you understand what is allowed and what is not in terms of using outside resources.

Remember that academic integrity is essential for your own personal and professional growth, and it is your responsibility to uphold these principles. So please take it seriously and always produce your own original work.