Fall 2024 CS 110: Computer Science Fundamentals
HW6 due on 11//2024 11:59 PM

1) Download the Python file Homework6-Exercises.py and work directly in this file only.
2) Please write your name, ID, and initial to the Honor Code statement.
3) Submit the python file as yourname_cs110_hw6.py and submit it to Canvas. Submit only this .py file.
4) Any late submission will be penalized 25% of the earned points per 24 hours.
5) Submissions must follow the instructions. Any incorrect format of submission (e.g., file, errors, etc.) is subject to be penalized.

Exercises

[20 pts] Question 1 - List Manipulation

Write a function named filter_odd_numbers that takes a list of integers as input and returns a new list that contains only the odd numbers from the original list.

Prompt the user to input a list of integers (comma-separated) and use the filter_odd_numbers function to display the odd numbers in the list.

Example Output:
Enter a list of integers: 1, 2, 3, 4, 5, 6
Odd numbers: [1, 3, 5]

[20 pts] Question 2: Weekday List Challenge

Create a list that represents the days of the week, with each day of the week as a string. Perform the following tasks:

1. **Create the list**: Initialize a list that contains the days of the week (from "Monday" to "Sunday").
2. **Access Days**: Print the third day in the list (hint: use index 2).
3. **Change a Day**: Change the last day in the list to "FunDay". Print the updated list.
4. **Add New Day**: Add "Holiday" to the end of the list and print the updated list.
5. **Remove a Day**: Remove "Wednesday" from the list and print the updated list.

Requirements:

- Use list indexing, modification, and the methods append() and remove().

[20 pts] Question 3: Create a Simple Bank Account Class

Write a class BankAccount that represents a simple bank account. The class should have the following:

1. **Attributes**:
   a. account_holder: The name of the account holder (string).
   b. balance: The current balance of the account (float).
2. **Methods**:
   a. deposit(amount): Adds the given amount to the balance.
   b. withdraw(amount): Subtracts the given amount from the balance (if there are enough funds).
   c. get_balance(): Returns the current balance.

Tasks:

1. Create a BankAccount for a person named "Alice" with an initial balance of 1000.
2. Call the deposit() method on the Alice object to add 500 to her balance.
3. Use the withdraw() method to subtract 200 from Alice's account.
4. Use the get_balance() method to print Alice's final balance.

[20 pts] Question 4: Movie Class

Create a class called Movie that represents a movie. The class should have the following attributes and methods:

Attributes:

- title: The title of the movie (string).
- director: The name of the director (string).
- year: The year the movie was released (integer).
- rating: The rating of the movie (float, between 1 and 10).

Methods:

- get_info(): Returns a string with the movie's title, director, year, and rating.
- update_rating(new_rating): Updates the rating of the movie.

Tasks:

1. Create a movie object with the title "Inception", directed by "Christopher Nolan", released in 2010, with a rating of 8.8.
2. Use the get_info() method to print out the details of the movie.
3. Use the update_rating() method to change the movie's rating to 9.2.
4. Print the updated details using get_info() again.

[20 pts] Question 5: The School Yearbook

Imagine you're working on a **digital yearbook** for a school. You have a table (a list of lists) that contains student information such as their name, grade, and favorite subject. Your task is to organize this data and perform a few operations on it.

Here is the students table, which contains the following information:

```
students = [
   ["Alice", 10, "Math"],   # Name, Grade, Favorite Subject
   ["Bob", 11, "History"],
   ["Charlie", 10, "Science"],
   ["Daisy", 12, "Math"],
   ["Eve", 11, "Art"]
]
```

Tasks:

1. **What is the grade of Charlie?**
   - Write a function find_grade(students, name) that takes the list of students and a student's name as input and returns the grade of that student. For example:

   find_grade(students, "Charlie") # Expected Output: 10

2. Write a function find_favorite_subject(students, subject) that takes the list of students and a subject as input and prints the names of students whose favorite

subject matches the input. If multiple students have the same favorite subject, print all their names. For example:

find_favorite_subject(students, "Art") # Expected Output: Eve

3. Write a function average_grade(students) that calculates and returns the average grade of all the students in the list.
