Midterm Lab Task 6.
Constructor Activity

Problem 1.

For this program, you are tasked to define the following:

Class - Money:

- Public Properties:
  - `amount` (type: int): Represents the monetary amount.
  - `denomination` (type: str): Specifies the denomination or currency type.
- Constructor:
  - `__init__(self, amount: int = 0, denomination: str = "Unknown")`:
    - This constructor can be used in three ways:
      - When called with no parameters, it initializes `amount` to 0 and `denomination` to "Unknown". This constructor is used when no specific monetary details are provided, setting default values.
      - When called with only the `amount` as a parameter, it sets the `amount` property accordingly and sets `denomination` to "Unknown". This constructor is useful when only the `amount` is known, but the `denomination` is not specified.
      - When called with both `amount` and `denomination` as parameters, it sets the respective properties to these values. This constructor is used when complete information about the monetary value, including its `denomination`, is available.

Note: Each class should be defined in its own file, with the file name following camelCase conventions (e.g., `bankAccount.py`).

Create a test class on a separate file named **testMoney.py**

Then try the sample output below:

classMoney.py

```python
class money(object):  7 usages

    def __init__(self, amount: int = 0, denomination: str = "Unknown"):
        self.amount = amount
        self.denomination = denomination

    def __str__(self):
        return f"{self.amount} {self.denomination}"

    def display(self):  3 usages
        if self.amount == 0 and self.denomination == "Unknown":
            print("Action: Invoking the Money class constructor using Money().")
            print("Output:")
            print(f"Amount: {self.amount}")
            print(f"Denomination: {self.denomination}\n")

        elif self.amount != 0 and self.denomination == "Unknown":
            print(f"Action: Invoking the Money class constructor using Money({self.amount}).")
            print("Output:")
            print(f"Amount: {self.amount}")
            print(f"Denomination: {self.denomination}\n")

        else:
            print(f'Action: Invoking the Money class constructor using Money({self.amount}, "{self.denomination}")')
            print("Output:")
            print(f"Amount: {self.amount}")
            print(f"Denomination: {self.denomination}\n")
```

testMoney.py

```python
from classMoney import money

def main():  1 usage
    mon1 = money()
    mon2 = money(200)
    mon3 = money( amount: 200 ,  denomination: "USD")

    money.display(mon1)
    money.display(mon2)
    money.display(mon3)

if __name__ == "__main__":
    main()
```

SAMPLE OUTPUT

```
Action: Invoking the Money class constructor using Money().
Output:
Amount: 0
Denomination: Unknown

Action: Invoking the Money class constructor using Money(200).
Output:
Amount: 200
Denomination: Unknown

Action: Invoking the Money class constructor using Money(200, "USD")
Output:
Amount: 200
Denomination: USD
```

Problem 2

classStudent.py

```python
class student():

    def __init__(self, id_number: int, name: str, course: str):
        self.id_number = id_number
        self.name = name
        self.course = course

    def __str__(self) -> str:
        return f"{self.id_number} - {self.name} - {self.course}"

    def validate_info(self):
        print("Action: Invoking validate_info() method with the following Student Information")
        print(f"ID: {self.id_number}")
        print(f"Name: {self.name}")
        print(f"Course: {self.course}")
        print("\nOutput:")

        if self.name.isalpha() and len(str(self.id_number)) > 9:
            print("Student information is valid.\n")
        else:
            print("Student information is not valid.\n")
```

testStudent.py

```python
from classStudent import student

def main():  1 usage

    st1 = student( id_number: 123456789,  name: "Jhon Doe",  course: "Computer Science" )
    st2 = student( id_number: 12345,  name: "Jane Doe",  course: "Mathematics")
    st3 = student( id_number: 987654321,  name: "Alice123",  course: "Physics")

    student.validate_info(st1)
    student.validate_info(st2)
    student.validate_info(st3)

if __name__ == '__main__':
    main()
```

Sample Output

```
Action: Invoking validate_info() method with the following Student Information
ID: 123456789
Name: Jhon Doe
Course: Computer Science

Output:
Student information is not valid.

Action: Invoking validate_info() method with the following Student Information
ID: 12345
Name: Jane Doe
Course: Mathematics

Output:
Student information is not valid.

Action: Invoking validate_info() method with the following Student Information
ID: 987654321
Name: Alice123
Course: Physics

Output:
Student information is not valid.
```