

Problem 2.

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

Class - Student:

- Public Properties:
 - `id_number` (type: int): A unique identifier for the student.
 - `name` (type: str): The name of the student.
 - `course` (type: str): The course the student is enrolled in.
- Methods:
 - `__str__()` -> `str`: Returns a string representation of the student's information in the format "{id_number} - {name} - {course}".
 - `validate_info()` -> `None`: Prints the message "Student information is valid." or "Student information is not valid." indicating whether the student's information is valid. Validity criteria include:
 - The `name` should contain only letters.
 - The `idNumber` should be exactly 9 digits long.

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 **testStudent.py**

CODE:

```
dunder.py ^ student.py ^ teststudent.py ^
4 usages
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 string in format:
        'id_number - name - course'"""
        return f"{self.id_number} - {self.name} - {self.course}"

3 usages
def validate_info(self) -> None:
    """Check if student info is valid"""
    if self.name.isalpha() and len(str(self.id_number)) == 9:
        print("Student information is valid.")
    else:
        print("Student information is not valid")
```

```
from student import Student
```

```
# Sample Output 1
```

```
print("Action: Invoking __str__() method"
```

```
      "with the following Student information:")
```

```
s1 = Student(id_number=123456789, name="JohnDoe", course="Computer Science")
```

```
print(f"ID: {s1.id_number}")
```

```
print(f"Name: {s1.name}")
```

```
print(f"Course: {s1.course}")
```

```
print(f"Output:")
```

```
print(s1)
```

```
s1.validate_info()
```

```
print("\n-----\n")
```

```
# Sample Output 2
```

```
print("Action: Invoking __str__() method"
```

```
      "with the following Student information:")
```

```
s2 = Student(id_number=12345, name="JaneDoe", course="Mathematics")
```

```
print(f"ID: {s2.id_number}")
```

```
print(f"Name: {s2.name}")
```

```
print(f"Course: {s2.course}")
```

```
print(f"Output:")
```

```
print(s2)
```

```
s2.validate_info()
```

```
print("\n-----\n")
```

```
# Sample Output 3
```

```
print("Action: Invoking __str__() method"
```

```
      "with the following Student information:")
```

```
s3 = Student(id_number=987654321, name="Alice123", course="Physics")
```

```
print(f"ID: {s3.id_number}")
```

```
print(f"Name: {s3.name}")
```

```
print(f"Course: {s3.course}")
```

```
print(f"Output:")
```

```
print(s3)
```

```
s3.validate_info()
```

OUTPUT:

```
teststudent x
C:\Users\COMLAB\AppData\Local\Programs\Python\Python311\python.exe C:\User
Action: Invoking __str__() methodwith the following Student information:
ID: 123456789
Name: JohnDoe
Course: Computer Science
Output:
123456789 - JohnDoe - Computer Science
Student information is valid.

-----

Action: Invoking __str__() methodwith the following Student information:
ID: 12345
Name: JaneDoe
Course: Mathematics
Output:
12345 - JaneDoe - Mathematics
Student information is not valid

-----

Action: Invoking __str__() methodwith the following Student information:
ID: 987654321
Name: Alice123
Course: Physics
Output:
987654321 - Alice123 - Physics
Student information is not valid

Process finished with exit code 0
```