Input/output and testing

5. Create two classes called `Customer` and `Purchase` that are designed to contain the contents of Table 1 and 2. Then add a list to the `Customer` class to hold `Purchase` objects.

Table 1: Customers

Id	Firstname	Surname
1	Amiee	Greene
2	Maia	Morley
3	Charleigh	Cano
4	Franklin	Torres
5	Mitchell	Page
6	Momina	Thornton
7	Cheryl	Devlin
8	Isobel	Orozco
9	Nicolas	Adams
10	Devante	Rodriguez

6. Use two CSV files that contain the values that are given in Table 1 and 2. Write a program to read the contents of two CSV files into `Customer` and `Purchase` objects. Each value of `CustomerId` in Table 2 corresponds to a value of `Id` in Table 1.

7. Add a `__repr__(self)` function to the `Customer` and `Purchase` classes to return the contents of the class as a string. Use these functions to test that the data are correctly read from the CSV files.

8. Add functions to the class `Customer` to:

   - Return a list of `ItemId` and `AmountPaid` by a specific customer. The return value should be a formatted string that includes the values and description of what they are.
   - Return a total `AmountPaid` by a specific customer. The return value should be a floating point number.

9. Draw a UML representation of the Customer and Purchase class, including association, attributes, operations and visibility.

10. Write a unit test program that:

    - Creates a `Customer` object and two `Purchase` objects.
    - Assigns the `Purchase` objects to the `Customer` object.
    - Calls the function to sum the `AmountPaid` by the customer.
    - Verifies that the result returned from the `AmountPaid` sum is correct.

Table 2: Purchases, where the foreign key `CustomerId` relates to the `Id` value in the Customers.

CustomerId	ItemId	AmountPaid
3	1	100
2	3	123
6	5	40
1	2	23
3	1	100
5	5	40
7	15	46
2	7	3.02
1	10	22
8	12	45.95
4	17	33
4	17	33
2	5	40