Project — Online Groceries

CS 3270

Background

With the COVID-19 quarantine, many people have ordered their groceries online for pickup or home delivery. This project will print a report of online orders from a fictional grocery chain. There are 3 input files to process:

- *customers.txt*
- *items.txt*
- *orders.txt*

The first file holds customer data in a single text line the following:

```
810003,Kai Antonikov,31 Prairie Rose Street,Philadelphia,PA,19196,215-975-7421,kantonikov0@4shared.com
```

The data fields are the customer id, name, street, city, state, zip, phone, and email.

The file *items.txt* has fields item_id, description, and price:

```
57464,Almonds Ground Blanched,2.99
```

Orders.txt holds **2 lines per order**. The first line contains the customer id, order number, order date, and then a variable-length sequence of `item_id-quantity` pairs:

```
762212,1,2020-03-15,10951-3,64612-2,57544-1,80145-1,27515-2,16736-1,79758-2,29286-2,51822-3,39096-1,32641-3, …
```

In the line above 3 items of product #10951 were ordered, 2 of product #64612, etc. Note that all fields in all files are comma-separated, and that the `item_id-quantity` pairs in *orders.txt* are separated by a dash.

The second line contains payment information in the form of payment code, and payment information. There are 3 types of payments:

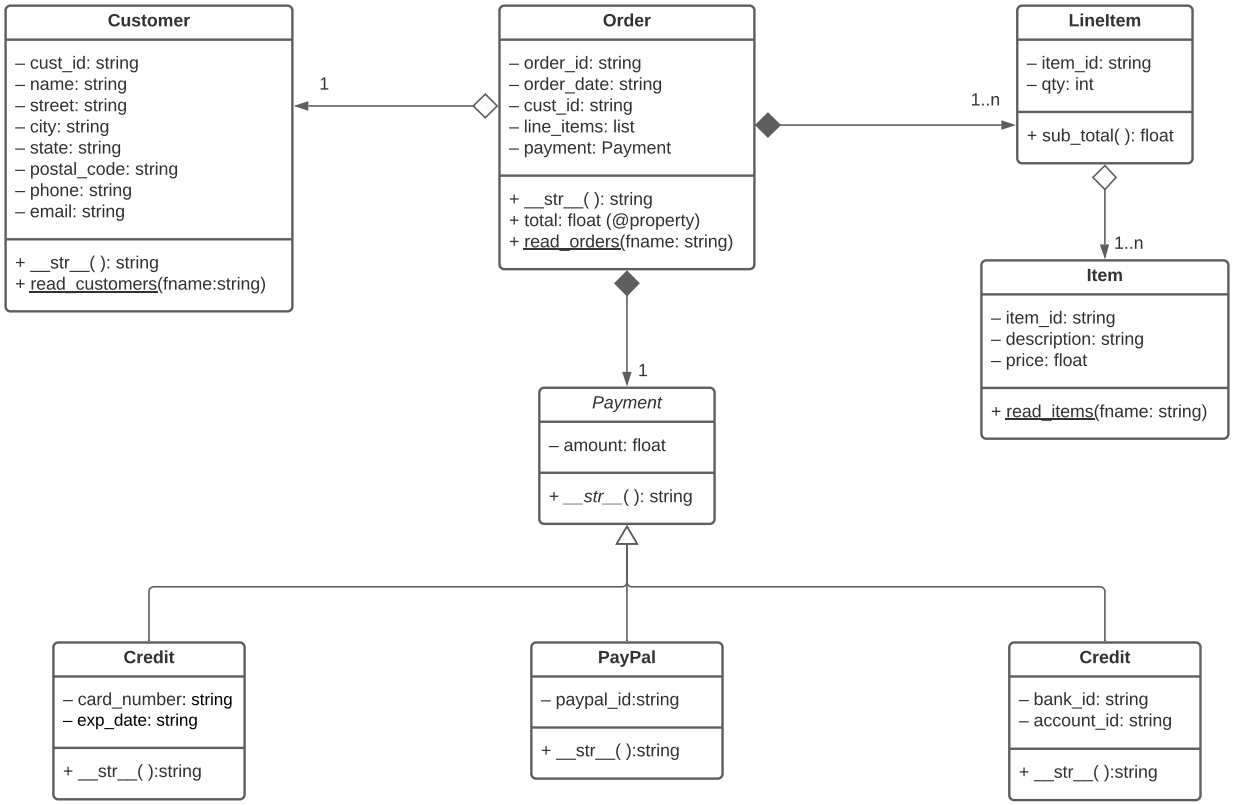
- 1 = Credit card (card number and expiration date)
- 2 = PayPal (paypal_id only)
- 3 = Wire transfer (bank_id and account_id)

Here are respective examples:

```
1,373975319551257,12-2023
2,cfeeham3s
3,72-2201515,68196-140
```

Requirements

The following UML class diagram suggests a design for this project.



Note that **Payment** should be an *abstract class* with abstract method **__str__**, which its derived classes must override to return a string which becomes part of printing an order, as illustrated in the sample output below. **Order.print_order** implicitly uses **Customer.__str__** and **Payment.__str__** to do its work.

Here is the **main** logic:

```
if __name__ == '__main__':
    Customer.read_customers("customers.txt")
    Item.read_items("items.txt")
    Order.read_orders("orders.txt")

    # Now print orders to the file order_report.txt:
    …
```

The static methods **Customer.read_customers** and **Item.read_items** create respectively global **list**s of **Customer** and **Item** objects obtained from file input. **Order.read_orders** populates a global list with Order objects obtained from file input.

Your output should be in a file named *order_report.txt* that looks like the following:

```
============================
Order #1, Date: 2020-03-15
Amount: $100.23, Paid by Credit card 373975319551257, exp. 12-2023

Customer ID #762212:
Yolanda McAlarney, ph. 505-136-7715, email: ymcalarney2u@wordpress.com
705 Corscot Hill
Albuquerque, NM 87190

Order Detail:
       Item 10951: "Syrup - Monin Swiss Choclate", 3 @ 3.00
       Item 16736: "Wine - Red Cooking", 1 @ 2.00
       Item 16974: "Pastry - Lemon Danish - Mini", 3 @ 1.19
       Item 23022: "Beef - Short Ribs", 1 @ 5.00
       Item 26461: "Bread - Crumbs Bulk", 1 @ 3.00
       Item 26860: "Waffle Stix", 2 @ 2.99
       Item 27515: "Longos - Burritos", 2 @ 4.00
       Item 29286: "Lemonade - Strawberry 591 Ml", 2 @ 2.25
       Item 32641: "Pie Filling - Cherry", 3 @ 0.89
       Item 39096: "Oil - Peanut", 1 @ 0.95
       Item 51822: "Pasta - Rotini Colour Dry", 3 @ 1.19
       Item 57544: "Peach - Halves", 1 @ 0.79
       Item 63725: "Black Currants", 3 @ 0.79
       Item 64007: "Juice - Propel Sport", 2 @ 1.99
       Item 64612: "Pie Shells 10", 2 @ 7.00
       Item 75536: "Quail - Whole Boneless", 2 @ 8.79
       Item 79758: "Beef Flat Iron Steak", 2 @ 5.49
       Item 80145: "Bread - Country Roll", 1 @ 2.29

============================
Order #2, Date: 2020-03-15
Amount: $74.48, Paid by Credit card 201741963232463, exp. 02-2022

Customer ID #258572:
Garey Baraja, ph. 260-560-6065, email: gbaraja5r@fda.gov
42 Kenwood Parkway
Fort Wayne, IN 46862

Order Detail:
       Item 18329: "Scallops - In Shell", 1 @ 4.49
       Item 21316: "Tomatoes - Roma", 3 @ 2.50
       Item 22248: "Basil - Seedlings Cookstown", 2 @ 3.89
       Item 30346: "Pasta - Orzo Dry", 1 @ 0.99
       Item 37948: "Shrimp - Black Tiger 16/20", 3 @ 7.60
       Item 47680: "Sandwich Wrap", 3 @ 2.00
       Item 56107: "Numi - Assorted Teas", 1 @ 5.99
       Item 56833: "Jolt Cola", 1 @ 1.29
       Item 59695: "Dr. Pepper - 355ml", 1 @ 1.59
       Item 63498: "Vinegar - Raspberry", 1 @ 1.79
       Item 66295: "Bread - Dark Rye Loaf", 1 @ 2.99
       Item 70616: "Lemonade - Black Cherry 591 Ml", 1 @ 2.89
       Item 80237: "Juice - Orange", 2 @ 2.19
       Item 96497: "Scampi Tail", 1 @ 4.00


...

============================
Order #8, Date: 2020-03-18
Amount: $113.63, Paid by Wire transfer from Bank ID 72-2201515, Account# 68196-140

Customer ID #217686:
Marisa Gossipin, ph. 239-305-6322, email: mgossipin1b@fema.gov
```

4 Charing Cross Lane
Fort Myers, FL 33994

Order Detail:
Item 11770: "Rice - Jasmine Sented", 1 @ 2.69
Item 12946: "Hagen Daza - Dk Choocolate", 1 @ 5.99
Item 17714: "Flour Pastry Super Fine", 3 @ 4.95
Item 21801: "Coffee - Ristretto Coffee Capsule", 2 @ 3.69
Item 23276: "Juice Peach Nectar", 1 @ 2.45
Item 24314: "Snapple - Mango Maddness", 1 @ 2.00
Item 39096: "Oil - Peanut", 2 @ 0.95
Item 39140: "Whmis Spray Bottle Graduated", 3 @ 1.99
Item 49318: "Green Scrubbie Pad H.duty", 3 @ 3.99
Item 54452: "Bananas", 2 @ 0.79
Item 55222: "Artichoke - Fresh", 2 @ 2.39
Item 59695: "Dr. Pepper - 355ml", 2 @ 1.59
Item 67408: "Pasta - Penne Primavera Single", 3 @ 1.49
Item 70461: "Pasta - Ravioli", 1 @ 1.25
Item 79084: "Red Cod Fillets - 225g", 2 @ 5.59
Item 79298: "Shiratamako - Rice Flour", 3 @ 3.49
Item 89924: "Towel - Roll White", 2 @ 2.19
Item 94416: "Shrimp - Baby Cold Water", 1 @ 4.99
Item 98317: "Stainless Steel Cleaner Vision", 3 @ 4.05

==========
Order #9, Date: 2020-03-18
Amount: $111.05, Paid by Paypal ID: tsantello5c

Customer ID #196547:
Alison Threader, ph. 703-698-2694, email: athreader5c@zimbio.com
50 Shopko Plaza
Washington, DC 20041

Order Detail:
Item 20755: "Sansho Powder", 2 @ 2.05
Item 21809: "Salmon - Atlantic Fresh Whole", 3 @ 4.75
Item 23022: "Beef - Short Ribs", 2 @ 5.00
Item 25956: "Energy Drink - Franks Pineapple", 3 @ 2.00
Item 37019: "Instant Coffee", 3 @ 6.00
Item 44214: "Wakami Seaweed", 1 @ 4.00
Item 47680: "Sandwich Wrap", 3 @ 2.00
Item 48704: "Beans - French", 1 @ 1.05
Item 51822: "Pasta - Rotini Colour Dry", 1 @ 1.19
Item 54044: "Sole - Fillet", 2 @ 3.99
Item 57544: "Peach - Halves", 1 @ 0.79
Item 64612: "Pie Shells 10", 2 @ 7.00
Item 67193: "Cheese - Cheddar Old White", 2 @ 1.88
Item 84418: "Soup - Knorr Country Bean", 2 @ 1.69
Item 90349: "Soup - Campbells Beef Noodle", 3 @ 1.19
Item 90475: "Chicken - Whole", 2 @ 6.49

...

==========
Order #609, Date: 2020-12-30
Amount: $101.69, Paid by Credit card 3538527630422753, exp. 04-2024

Customer ID #409485:
Jaye Martinho, ph. 361-111-4183, email: jmartinho7x@dion.ne.jp
2339 Magdeline Plaza
Corpus Christi, TX 78410

Order Detail:
Item 12527: "Bagelers - Cinn / Brown Sugar", 3 @ 3.99
Item 12568: "Cabbage - Red", 2 @ 0.69

```
Item 16724: "Blueberries", 3 @ 2.99
Item 17981: "Halibut - Fletches", 1 @ 8.00
Item 32641: "Pie Filling - Cherry", 3 @ 0.89
Item 41142: "Vinegar - Red Wine", 3 @ 1.25
Item 44214: "Wakami Seaweed", 1 @ 4.00
Item 56826: "Flour - Bread", 2 @ 4.00
Item 57464: "Almonds Ground Blanched", 2 @ 2.99
Item 58524: "Thyme - Lemon Fresh", 1 @ 2.19
Item 64067: "Magnotta Bel Paese Red", 2 @ 3.39
Item 67408: "Pasta - Penne Primavera Single", 3 @ 1.49
Item 78265: "Syrup - Monin Irish Cream", 3 @ 3.00
Item 80145: "Bread - Country Roll", 2 @ 2.29
Item 81169: "Table Cloth 90x90 White", 1 @ 3.99
Item 86456: "Rice - Long Grain", 1 @ 2.29
Item 86494: "Crackers - Trio", 2 @ 2.35
Item 95662: "Garlic Powder", 3 @ 2.99
```

Each order begins with a dashed line and has 3 parts, each followed by an empty line. The orders appear in the order they are read from the file (they are in date order), but the items in each order are sorted by **item_id**.

Submit your *groceries.py* and *order_report.txt* files in Canvas by the due date.

Implementation Notes

- Recommend using `@dataclass` for Customer, **Item**, **LineItem**, and **Order**.
- **Customers** and **Item** objects are found as needed by their respective IDs; don't duplicate data.
- Make **Order.total** a read *property*; it returns **payment.amount**. Set the payment amount in **read_orders**.
- Note the formatting of the report, including 2 decimals for the order total. Match this output format exactly.
- You may have seen this project in CS 2370. The Python version is over 100 lines shorter. Make your solution Pythonic.