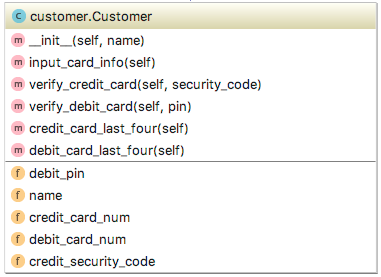
## Programming Project

### Instructions – Customer class (customer.py)

Create a Python project for the following problem. Zip the Python project into a zip file. Submit the zip file to Blackboard for credit.

In Lab 12 you wrote a program for Wake-Mart customers to check out grocery items. Now Wake-Mart wants an upgrade of the system. First, they want the customer to register before ordering items. They only accept payment by credit card or debit card. The customer must enter card information during the registration process. Second, instead of asking the customer to enter the price of each item and the value of each coupon, now they want the customer to enter a 4-digit code for each item and each coupon, and the program will look up the price from a price list or the value of the coupon from a coupon list. The following lists the requirements of this project.

Define a Customer class. The following UML diagram generated by PyCharm shows the design of this class:



The class has six methods:

1. \_\_init\_\_ – The constructor receives customer name as an argument and stores it in an instance variable. It should also create the other instance variables and initialize them all to empty strings.
2. input\_card\_info – In this method, the user enters credit card number, credit card security code, debit card number and debit card PIN. All these input values should be stored in instance variables.
3. verify\_credit\_card – This method will check whether the security code is correct. This method is called when the user chooses to pay by credit card. It compares the security code entered by the user during check out against the security code stored in the Customer object. If they are the same, this method returns True. Otherwise, it returns False.
4. verify\_debit\_card – This method is similar to verify\_credit\_card. It compares the PIN entered by the user during checkout against the PIN stored in the Customer object. If they are the same, this method returns True. Otherwise, it returns False.
5. credit\_card\_last\_four method returns the last four digits of the credit card number.
6. debit\_card\_last\_four returns the last four digits of the debit card number.

This class has five private instance variables to store customer name, credit card number, credit card security code, debit card number, and debit card PIN.

### Instructions – Main Module (final-exam.py)

The main module of the project must have the following six functions.

1. read\_price\_list – The price list is stored in the text file “price\_list.txt”. Each line in this file contains a 4-digit product code and unit price of an item. Write a read\_price\_list function to read the price list into the program. Store the data in a dictionary with product codes as keys and prices as values. Return the dictionary. The file “price\_list.txt” has been created for you. You just need to download it from Blackboard.
2. read\_coupon\_list – The coupon list is stored in the text file “coupon\_list.txt”. Each line in this file contains a 4-digit code and monetary value of a coupon. Write a read\_coupon\_list function to read the coupon list into the program. Store the data in a dictionary with the 4-digit codes as keys and monetary values as values. Return the dictionary. The file “coupon\_list.txt” has been created for you. You just need to download it from Blackboard.
3. scan\_prices – Write a scan\_prices function for the customer to order items. First call the read\_price\_list function to read the price list from a text file. Display all items in the price dictionary. Then use a loop for the customer to enter product code of each item he wants. Every time a product code is entered, check to see whether the code is in the dictionary. If it is not, display “Item not found”. If the code is in the dictionary, display “Item found” and the price of the item. When the customer has no more product code to enter, she types ‘9999’ to exit the loop. The scan\_prices function will calculate, display and return the total price of all the items ordered.
4. scan\_coupons – Write a scan\_coupons function for the customer to enter coupons. This function is very similar to the scan\_prices function. It calls the read\_coupon\_list function to read the coupon list from a text file. Display all items in the coupon dictionary. Then use a loop for the customer to enter 4-digit code of each coupon she has. Every time a 4-digit code is entered, check to see whether the code is in the dictionary. If it is not, display “Coupon not found.” If the code is in the dictionary, display “Coupon found” and the value of the coupon. When the customer has no more coupons to enter, she types ‘9999’ to exit the loop. The scan\_coupons function will calculate, display and return the total value of all the coupons entered.
5. make\_payment – The make\_payment function has two parameters to receive the Customer object and the total amount to pay. It asks the customer to choose a payment method. If the user chooses to pay by credit card, ask the user to enter the security code. Call the verify\_credit\_card method of the Customer object to verify the security code. If the security code is incorrect, display “Security code incorrect. Payment rejected.” and loop back to ask the customer to choose payment method again. If the security code is correct, display on the screen the amount to be charged and the last four digits of the credit card number.

If the user chooses to pay by debit card, ask the user to enter the PIN. Call the verify\_debit\_card method of the Customer object to verify the PIN. If the PIN is incorrect, display “PIN incorrect. Payment rejected.” and loop back to ask the customer to choose payment method again. If PIN is correct, display on the screen the amount to be charged and the last four digits of the debit card number.

1. main – In the main function:
   1. Ask the customer to enter her name. Use it to create a Customer object.
   2. Call the input\_card\_info method of the object to input credit card and debit card information.
   3. Call the scan\_prices function.
   4. Call the scan\_coupons function.
   5. Use the scan\_prices and scan\_coupons return values to calculate how much the customer needs to pay.
   6. Display this amount on the screen.
   7. Call the make\_payment method to process payment. Pass the Customer object and the total amount to pay as two arguments.

### Sample Output

Welcome to Wake-Mart. Please register.

Enter your name: Clark Kent

Enter credit card number: 1234-5678-1234-5678

Enter 3-digit security code: 111

Enter debit card number: 9876-5432-9876-5432

Enter 4-digit PIN: 2222

Registration completed

Price list:

('4178', 8.0)

('6245', 4.88)

('5206', 4.25)

('2159', 0.99)

('6625', 2.99)

('1423', 7.88)

('1752', 4.99)

('6112', 5.77)

('2487', 12.52)

('2368', 5.25)

Enter 4-digit item code [or 9999 to stop]: 1243

Item not found

Enter 4-digit item code [or 9999 to stop]: 1423

Item found. Price: 7.88

Enter 4-digit item code [or 9999 to stop]: 2487

Item found. Price: 12.52

Enter 4-digit item code [or 9999 to stop]: 2368

Item found. Price: 5.25

Enter 4-digit item code [or 9999 to stop]: 9999

Total price: 25.65

Coupon list:

('7468', 1.99)

('8546', 1.29)

('7612', 1.49)

('7125', 0.99)

('8207', 0.59)

Enter 4-digit coupon code [or 9999 to stop]: 7162

Coupon not found

Enter 4-digit coupon code [or 9999 to stop]: 8207

Coupon found. Value: 0.59

Enter 4-digit coupon code [or 9999 to stop]: 7468

Coupon found. Value: 1.99

Enter 4-digit coupon code [or 9999 to stop]: 9999

Total coupon value: 2.58

Please pay this amount: 23.07

Please choose payment method.

Please enter 1 for credit card, 2 for debit card: 1

Please enter 3-digit security code: 222

Security code incorrect. Payment rejected.

Please choose payment method.

Please enter 1 for credit card, 2 for debit card: 2

Please enter 4-digit PIN: 1111

PIN incorrect. Payment rejected.

Please choose payment method.

Please enter 1 for credit card, 2 for debit card: 1

Please enter 3-digit security code: 111

The amount of 23.07 will be charged to card number ending with 5678

### Grading

Zip your Python project and submit it to Blackboard for credit. Your Python project should have at least four files: a Python file containing the definition of Customer class, a Python file containing the main module, the text file price\_list.txt, and the text file coupon\_list.txt.

* \_\_init\_\_ method of Customer class [10 points]
* input\_card\_info method of Customer class [10 points]
* verify\_credit\_card method of Customer class [5 points]
* verify\_debit\_card method of Customer class [5 points]
* credit\_card\_last\_four method of Customer class [5 points]
* debit\_card\_last\_four method of Customer class [5 points]
* main function [10 points]
* read\_price\_list function [10 points]
* read\_coupon\_list function [10 points]
* scan\_prices function [10 points]
* scan\_coupons function [10 points]
* make\_payment function [10 points]