WIA1002 Data Structure
Lab 6: Queue

1. Write a program to read the transactions input from a string into a queue. Given D is deposit and W is withdraw. After that, show the balance for each transaction, given the initial balance is 500. Display error if the user withdraws more than the balance.

   Example output:

```
Enter transactions : D 400 | W 300 | W 700 | D 200 | D 450 | W 120
D 400  -->  W 300  -->  W 700  -->  D 200  -->  D 450  -->  W 120 -->
Initial Balance : 500
Deposit 400              New Balance 900
Withdraw 300             New Balance 600
Withdraw 700 Rejected    New Balance 600
Deposit 200              New Balance 800
Deposit 450              New Balance 1250
Withdraw 120             New Balance 1130
```

2. Create a program to read data from a text file (**lab6Q2.txt**) and then categorize the product into different queues based on the product code. The number of queue is variable and is determined by reading the data from the text file.

   Contents of Text File:

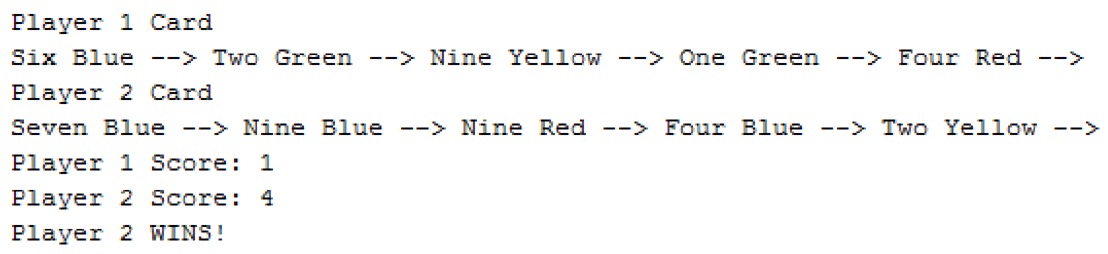
   P03 Durian P02 iPhone P06 Buku P02 Samsung P04 Baju P03 Tembikai
   P03 Mangga P06 Pembaris P02 Nokia P06 Kertas P04 Kasut P03 Rambutan

   Example output:

```
Product Code in Queue : P03 --> P02 --> P06 --> P04 -->
List of product by categories
Product : P03
Durian --> Tembikai --> Mangga --> Rambutan -->
Product : P02
iPhone --> Samsung --> Nokia -->
Product : P06
Buku --> Pembaris --> Kertas -->
Product : P04
Baju --> Kasut -->
```

3. Colour Card is a new type of card that consists of 4 different colour (Blue, Green, Red, Yellow) from one to ten. The total number of card is 40. Create a program that simulate the two players Colour Card game where each player will draw 5 cards from the deck in sequence and then compare the cards in sequence with each other. The card with the higher number is the bigger card and if the card is the same number, the Blue colour is the bigger followed by Green, Red, Yellow. Create the ColourCard class and then use different queues for different players.

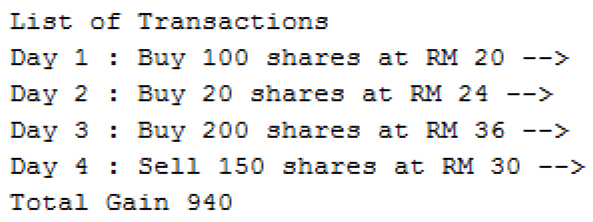
Example output:



```
Player 1 Card
Six Blue --> Two Green --> Nine Yellow --> One Green --> Four Red -->
Player 2 Card
Seven Blue --> Nine Blue --> Nine Red --> Four Blue --> Two Yellow -->
Player 1 Score: 1
Player 2 Score: 4
Player 2 WINS!
```

4. When a share of common stock of some company is sold, the capital gain or loss is obtained by calculating the difference between the share's selling price and the price originally paid to buy it. This rule is easy to understand for a single share. However, if we sell multiple shares of stock bought over a long period of time, then we must identify the shares actually being sold. A standard accounting principle for identifying which shares of a stock were sold in such a case is to use FIFO – the shares sold are the ones that have been held the longest. For example, we buy 100 shares at RM20 each on day 1, 20 shares at RM24 on day 2, 200 shares at RM36 on day 3, and then sell 150 shares on day 4 at RM30 each. Using FIFO, the capital gain or loss is (100 * (30-20)) + (20 * (30-24)) + (30 * (30-36)) = RM940. Write a program to read the transactions from a text file (**lab6Q4.txt**), compute the total gain or loss.

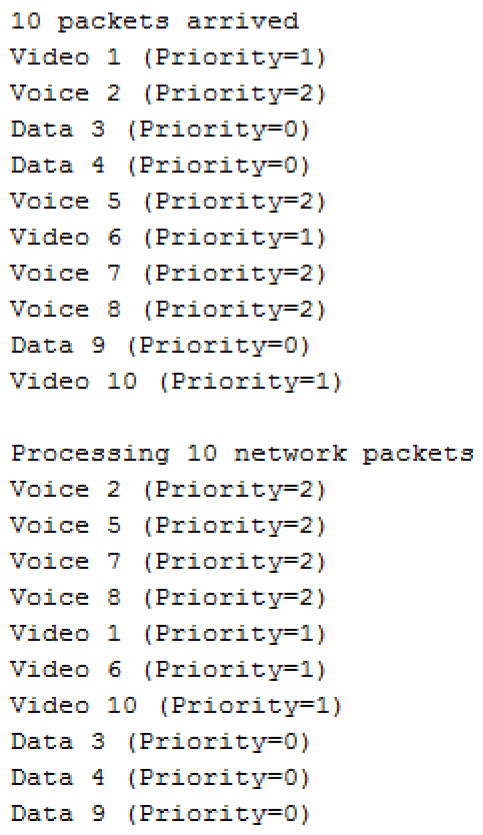
   Example output:



```
List of Transactions
Day 1 : Buy 100 shares at RM 20 -->
Day 2 : Buy 20 shares at RM 24 -->
Day 3 : Buy 200 shares at RM 36 -->
Day 4 : Sell 150 shares at RM 30 -->
Total Gain 940
```

5. Create a program to simulate the processing of network packet. The network devices forward the packet according to the type of packet. The voice packet has highest priority as compared to video packet and data packet. Create a Packet class that consists of the type of packet and the priority. After that, create a PriorityQueue class that consists of all the methods from the Queue class. Then, modify the enqueue method to allow the highest priority packet to queue first and the dequeue method to allow the highest priority packet to process first.

   Example output:



```
10 packets arrived
Video 1 (Priority=1)
Voice 2 (Priority=2)
Data 3 (Priority=0)
Data 4 (Priority=0)
Voice 5 (Priority=2)
Video 6 (Priority=1)
Voice 7 (Priority=2)
Voice 8 (Priority=2)
Data 9 (Priority=0)
Video 10 (Priority=1)

Processing 10 network packets
Voice 2 (Priority=2)
Voice 5 (Priority=2)
Voice 7 (Priority=2)
Voice 8 (Priority=2)
Video 1 (Priority=1)
Video 6 (Priority=1)
Video 10 (Priority=1)
Data 3 (Priority=0)
Data 4 (Priority=0)
Data 9 (Priority=0)
```