

CPE217 – Homework 4

Homework: Realized and Unrealized Profits and Losses

Homework Due Date: 22 September 2019

Patiwet Wuttisarnwattana, Ph.D.

Department of Computer Engineering

- คำชี้แจงการส่งงาน
- แต่ละกลุ่ม ควรให้ core person เป็นคนส่งงาน และในช่องข้อความต้องระบุรหัสประจำตัวนักศึกษาของทุกคนที่เป็นสมาชิกในกลุ่ม หาก Core Person ไม่สามารถส่งงานได้ ให้สมาชิกคนใดก็ได้ส่งงานแทน แต่ต้องบอกว่าส่งแทน Core Person ซึ่งก็คือใคร มีรหัสอะไร
- โค้ดของคุณต้องมีคอมเมนต์ (comment) เพื่ออธิบายว่าโค้ดดังที่เห็นอยู่นี้ทำงานอะไร หรือ if นี่ทำตรวจสอบอะไร หากกลุ่มไหนไม่มีคอมเมนต์ในโค้ดจะไม่ได้รับการตรวจ การเขียนคอมเมนต์ไม่ต้องเขียนแบบละเอียดยิบก็ได้เท่าที่คุณต้องการให้ผู้ตรวจทราบก็พอ
- งานที่ส่งต้องประกอบด้วย ZIP file ของ src folder ที่สามารถกด F6 รันได้เลย หากมี compile error หรือ runtime exception งานของนักศึกษาจะไม่ได้รับการตรวจ
- สามารถส่งการบ้านช้าได้ แต่หักคะแนนวันละ 10%
- การลอกงานเพื่อนมาส่ง เป็นการทุจริตและมีความผิดทางวินัย หากตรวจพบอาจารย์อาจพิจารณาให้คะแนนการบ้านนั้น หรือคะแนนการบ้านทั้งหมดของคุณ และ/หรือ คะแนนจิตพิสัย ทั้งหมดได้ศูนย์คะแนน ซึ่งนั่นอาจเป็นปัจจัยของการตัดสินใจถอนกระบวนวิชานี้ของคุณและลงทะเบียนใหม่ในปีการศึกษาหน้า
- การลอกงานมาส่งต้องรับผิดชอบพร้อมกันทั้งกลุ่ม จะให้คนทำผิด รับผิดชอบแต่เพียงคนเดียวไม่ได้
- เพื่อนในกลุ่มที่เหลือมีหน้าที่ต้องตรวจสอบความถูกต้องและรับประกันผลงานว่าไม่นำผลงานของกลุ่มอื่นมาส่ง

การบ้านนี้นักศึกษาจะได้เรียนรู้การประยุกต์ใช้ Queue กับ Stack ในระบบบัญชีของบริษัทหลักทรัพย์ ในรายงานผลการลงทุนในหุ้นมักจะรายงานกำไรขาดทุน (Profit/Loss) อยู่สองประเภท ประเภทแรกคือ กำไร/ขาดทุนทางบัญชี (Unrealized Profit/Loss) และประเภทที่สองคือ กำไร/ขาดทุนที่เกิดขึ้นจริง (Realized Profit/Loss) สำหรับกำไร/ขาดทุนทางบัญชี (Unrealized Profit/Loss) คือ การที่นักลงทุนซื้อหุ้นที่ราคาหนึ่ง ๆ ต่อมามูลค่าหุ้นเกิดการเปลี่ยนแปลงไปตามกลไกของตลาด ทำให้เกิดกำไร/ขาดทุนทางบัญชี (โดยยังไม่ได้ขายหุ้นนั้นจริง ๆ) เช่น เมื่อวานนักลงทุนซื้อหุ้นบริษัท ABC มา 10 หุ้น ที่ราคา 100 บาทต่อหุ้น วันนี้ตลาดรับซื้อหุ้น ABC อยู่ที่ราคา 120 บาทต่อหุ้น ดังนั้นนักลงทุนก็จะมี กำไรทางบัญชี (Unrealized Profit) คิดจาก $(\text{ราคาหุ้นปัจจุบัน} - \text{ราคาต้นทุน}) \times \text{จำนวนหุ้น} = (120 - 100) \times 10 = 200$ บาท トラバิดที่นักลงทุนยังไม่ขายหุ้นออกไป กำไรนี้ก็ยังเป็นกำไรทางบัญชีอยู่เพราะยังไม่ได้เกิดขึ้นจริง ราคาอาจปรับขึ้นหรือปรับลงในอนาคตต่อไปได้ แต่ถ้านักลงทุนขายหุ้นนี้เมื่อไหร่ กำไร 200 บาทนี้ก็จะเรียกว่ากำไรที่เกิดขึ้นจริงทันที (Realized Profit) ทันที ส่วนการคิดการขาดทุนของการลงทุน (Loss) ก็เช่นเดียวกัน ต่างกันตรงที่ตัวเลขคำนวณได้จะติดลบเพราะซื้อมาที่ราคาสูงแต่ขายที่ราคาต่ำจึงเป็นการขาดทุน เช่นเมื่อวานซื้อหุ้น XYZ มา 10 หุ้น ที่ราคา 120 บาทต่อหุ้น วันนี้ราคาตกเหลือ 100 บาทต่อหุ้น นักลงทุนจึงเกิดการขาดทุนทางบัญชี (Unrealized Loss) ทันทีที่ $(100 - 120) \times 10 = -200$ บาท หากนักลงทุนตัดสินใจขายที่ราคานี้ นักลงทุนก็จะเกิดการขาดทุนจริง (Realized Loss) ที่ -200 บาท

หากมีการซื้อขายหุ้นหลาย ๆ รายการ ดังตัวอย่าง: ในเดือนกันยายน 2560 นักลงทุนทำรายการซื้อขายหุ้น ABC ทั้งหมด 5 รายการ ดังนี้

วันที่	ทำการ	จำนวนหุ้น	ราคาซื้อ/ราคาขาย (บาทต่อหุ้น)	จำนวนหุ้นที่มีในบัญชีทั้งหมด เมื่อสิ้นวันทำการ
1/9/2560	ซื้อ	+10	100	10
2/9/2560	ซื้อ	+10	150	20
3/9/2560	ซื้อ	+20	110	40
4/9/2560	ซื้อ	+20	160	60
5/9/2560	ขาย	-25	130	35

- คำถามคือ เมื่อนักลงทุนขายหุ้นออก 25 หุ้นในวันสุดท้าย เขามีกำไร/ขาดทุนเป็นอย่างไร?

โดยหลักการคิดนั้นจะมีอยู่สองวิธี คือ FIFO (First In First Out) หรือ LIFO (Last In First Out)

หากใช้หลักการแรก FIFO ในการคิดแล้ว หุ้น ABC 25 หุ้นที่ขายออกวันที่ห้านี้ 10 หุ้นที่ซื้อมาวันแรกจะต้องออกไปก่อน; 10 หุ้นที่ซื้อมาวันที่สองก็ต้องออกไปหมดเช่นกัน; ส่วนหุ้นที่ซื้อวันที่สามจะออกไปแค่ 5 หุ้น รวมขายหุ้นทั้งหมด 25 หุ้น (เหลือค้างบัญชีอีก 35 หุ้น) เพราะฉะนั้น กำไรที่เกิดขึ้นจริง (Realized Profit) จึงคิดตามรายการทั้งสาม ดังนี้ $(130 - 100) \times 10 + (130 - 150) \times 10 + (130 - 110) \times 5 = 200$ บาท ส่วนกำไร/ขาดทุนทางบัญชี (Unrealized Profit/Loss) ให้คิดจากหุ้นที่เหลืออยู่ คือ $(130 - 110) \times 15 + (130 - 160) \times 20 = -300$ (ซึ่งอยู่ในสถานะของการขาดทุนทางบัญชี หรือ Unrealized Loss เพราะตัวเลขติดลบ)

หากใช้หลักการที่สอง LIFO หุ้น ABC 25 หุ้นที่ขายออกวันที่ห้าวัน 20 หุ้นที่ซื้อในวันล่าสุด (วันที่สี่) จะต้องออกไปทั้งหมดก่อน; อีก 5 หุ้นจึงเอามาจากที่ซื้อวันล่าสุดถัดไป นั่นก็คือวันที่สาม เพราะฉะนั้น กำไรขาดทุนที่เกิดขึ้นจริง (Realized Profit/Loss) ที่คิดด้วยวิธี LIFO จึงมีค่าเท่ากับ $(130-160) \times 20 + (130-110) \times 5 = -500$ มีค่าติดลบแบบนี้จึงเรียกว่า Realized Loss ส่วน กำไรขาดทุนทางบัญชีก็คิดจากหุ้นที่เหลือ คือ $(130-110) \times 15 + (130-150) \times 10 + (130-100) \times 10 = 400$ มีค่าเป็นบวกแบบนี้จึงเรียกว่า Unrealized Profit

จากความรู้ที่อาจารย์อธิบายมานี้ ให้นักศึกษาเขียนโปรแกรมเพื่อคำนวณกำไร/ขาดทุน ทั้งทางบัญชีและที่เกิดขึ้นจริง ทั้งวิธีคิดแบบ FIFO (Queue Data Structure) และ LIFO (Stack Data Structure)

ให้นักศึกษาทำความเข้าใจ Starter Code ของอาจารย์ที่เตรียมไว้ให้ ซึ่งมีรายละเอียดคร่าว ๆ ดังต่อไปนี้

- Class Node ทำหน้าที่ บรรจุข้อมูลหุ้นที่ซื้อในแต่ละวัน ซึ่งประกอบด้วยข้อมูลจำนวนหุ้นที่ซื้อ (int shares) และราคาต่อหุ้น ณ เวลาซื้อ (double price) อาจารย์ได้เขียนโค้ดไว้ให้เสร็จสมบูรณ์แล้ว ไม่ต้องแก้ไขอะไรเพิ่มเติม
- Class Queue ทำหน้าที่เป็นโครงสร้างข้อมูล Queue ที่สามารถบรรจุ Node ได้ตามหลักการ FIFO ที่เรียนในห้อง โดยอาจารย์กำหนดให้ Queue นี้ ต้องสร้างโดยใช้ Singly Linked List with Tail เท่านั้น มีฟังก์ชันที่เกี่ยวข้องคือ
 - `public void push(Node node)` ทำหน้าที่ นำ Node เข้าต่อด้านหลังของคิว
 - `public void pop()` ทำหน้าที่ นำ Node ที่อยู่ด้านหน้าสุดของคิวออกจากคิว โดยไม่ต้อง return Node นั้นออกมา
 - `public Node top()` ทำหน้าที่ return Node ที่อยู่ด้านหน้าสุดของคิวให้กับ Caller โดยไม่ต้อง Pop Node นั้นออกมา
 - ฟังก์ชันทั้งสามนี้ ไม่สมบูรณ์ คุณต้องแก้ไข/เพิ่มเติมโค้ดให้สามารถทำงานได้
- Class Stack ทำหน้าที่เป็นโครงสร้างข้อมูล Stack ที่สามารถบรรจุ Node ได้ตามหลักการ LIFO ที่เรียนในห้อง โดยอาจารย์กำหนดให้ Stack นี้ ต้องสร้างโดยใช้ Singly Linked List without Tail เท่านั้น มีฟังก์ชันที่เกี่ยวข้องคือ
 - `public void push(Node node)` ทำหน้าที่ นำ Node เข้าต่อด้านบนของสแต็ก
 - `public void pop()` ทำหน้าที่ นำ Node ที่อยู่ด้านบนสุดของสแต็กออกจากสแต็ก โดยไม่ต้อง return Node นั้นออกมา
 - `public Node top()` ทำหน้าที่ return Node ที่อยู่ด้านบนสุดของสแต็กให้กับ Caller โดยไม่ต้อง Pop Node นั้นออกมา
 - ฟังก์ชันทั้งสามนี้ ไม่สมบูรณ์ คุณต้องแก้ไข/เพิ่มเติมโค้ดให้สามารถทำงานได้
- Class Stock ทำหน้าที่ประมวลผลการซื้อขายหุ้นตามที่อาจารย์อธิบายให้ข้างต้น โดยมีฟังก์ชันที่เกี่ยวข้องดังต่อไปนี้
 - `public Stock(String costBasis)` ทำหน้าที่ สร้าง Object ของ Queue หรือ Stack ตามที่ผู้ใช้งานกำหนดเข้ามาผ่าน costBasis ฟังก์ชันนี้อาจารย์เขียนโค้ดให้เสร็จสมบูรณ์แล้ว ไม่ต้องแก้ไขเพิ่มเติม

- `public void buy(int boughtShares, double boughtPrice)` ทำหน้าที่ ซื้อหุ้นเข้าบัญชี ตามจำนวนและราคาที่ระบุไว้ใน `boughtShares` และ `boughtPrice` ตามลำดับ หน้าที่ของคุณคือ ต้องเอาจำนวนหุ้นและราคาเข้าไปบรรจุไว้ใน Node แล้ว Push Node ดังกล่าวเข้าไปเก็บไว้ใน Data structure ที่เหมาะสมตาม cost basis (FIFO or LIFO) โค้ดส่วนนี้แก้ไขเพิ่มเติมคนเดียวอาจารย์ใบให้ คุณเพิ่มแค่หนึ่งบรรทัด
- `public void sell(int soldShares, double soldPrice)` ทำหน้าที่ ขายหุ้นตามคำสั่ง โดยจำนวนขายและราคาขาย เป็นไปตามที่ระบุไว้ใน `soldShares` และ `soldPrice` ตามลำดับ หน้าที่ของคุณคือ ต้องเอาข้อมูลหุ้นที่บรรจุใน Node ซึ่งอยู่ใน Data structure ออกมา (Top หรือ Pop) แล้วทำการคำนวณ Realized Profit/Loss และ Unrealized Profit/Loss ตามที่อาจารย์อธิบายไปก่อนหน้านี้
 - ทุก ๆ ครั้งที่มีการส่งคำสั่ง Sell ต้องมีการรายงานค่า Profit/Loss ออกมาทาง System.out แต่ถ้ามีแต่คำสั่ง Buy ไม่ต้องรายงานใด ๆ ส่วนนี้อาจารย์ก็เขียนไว้ให้แล้ว
 - หากจำนวนหุ้นในคำสั่งขาย มีมากกว่าจำนวนหุ้นที่ซื้อเข้ามา ให้แจ้ง Error แก่ User ว่า Sell command rejected แล้วไม่ต้องทำอะไร ส่วนนี้อาจารย์ก็เขียนไว้ให้แล้วเช่นกัน
 - โค้ดส่วน Sell นี้ค่อนข้างมีความซับซ้อนพอสมควร เป็นจุดที่ต้องใช้เวลามากที่สุด หากไม่เข้าใจคำถามหรืออัลกอริทึมในการคิด ขอให้รับมาปรึกษาอาจารย์เป็นการด่วน อย่าเสียเวลางม เอาเวลาไปอ่านหนังสือเตรียมสอบมิดเทอมดีกว่า
- `public void showList()` ทำหน้าที่ แสดง Node ที่ต่ออยู่ใน List ว่ามี Node อะไรบ้าง ส่วนนี้อาจารย์เขียนไว้ให้แล้ว ไม่ต้องแก้ไขเพิ่มเติมอะไร

ตัวอย่างการทำงาน

Java code
<pre>public static void main(String[] args) { Stock ptt = new Stock("FIFO"); ptt.showList(); ptt.buy(10, 100); ptt.buy(10, 150); ptt.showList(); ptt.buy(20, 110); ptt.buy(20, 160); ptt.showList(); }</pre>
Output
<pre>head -> tail head -> [10@100.0B] -> [10@150.0B] -> tail head -> [10@100.0B] -> [10@150.0B] -> [20@110.0B] -> [20@160.0B] -> tail</pre>

Java code
<pre> public static void main(String[] args) { Stock ptt = new Stock("FIFO"); ptt.buy(10, 100); ptt.buy(10, 150); ptt.buy(20, 110); ptt.buy(20, 160); ptt.showList(); ptt.sell(25, 130); ptt.showList(); } </pre>
Output
<p>head -> [10@100.0B] -> [10@150.0B] -> [20@110.0B] -> [20@160.0B] -> tail</p> <p>Realized P/L = 200.0 Unrealized P/L = -300.0</p> <p>head -> [15@110.0B] -> [20@160.0B] -> tail</p>

Java code
<pre> public static void main(String[] args) { Stock ptt = new Stock("LIFO"); ptt.buy(10, 100); ptt.buy(10, 150); ptt.buy(20, 110); ptt.buy(20, 160); ptt.showList(); ptt.sell(25, 130); ptt.showList(); } </pre>
Output
<p>head -> [20@160.0B] -> [20@110.0B] -> [10@150.0B] -> [10@100.0B] -> tail</p> <p>Realized P/L = -500.0 Unrealized P/L = 400.0</p> <p>head -> [15@110.0B] -> [10@150.0B] -> [10@100.0B] -> tail</p>

Java code

```
public static void main(String[] args) {  
    Stock cpall = new Stock("FIFO");  
    cpall.buy(35, 100);  
    cpall.sell(25, 130);  
    cpall.buy(5, 130);  
    cpall.sell(10, 120);  
    cpall.sell(5, 120);  
  
    cpall = new Stock("LIFO");  
    cpall.buy(35, 100);  
    cpall.sell(25, 130);  
    cpall.buy(5, 130);  
    cpall.sell(10, 120);  
    cpall.sell(5, 120);  
}
```

Output

Realized P/L = 750.0 Unrealized P/L = 300.0

Realized P/L = 200.0 Unrealized P/L = -50.0

Realized P/L = -50.0 Unrealized P/L = 0.0

Realized P/L = 750.0 Unrealized P/L = 300.0

Realized P/L = 50.0 Unrealized P/L = 100.0

Realized P/L = 100.0 Unrealized P/L = 0.0

Java code

```
public static void main(String[] args) {  
    Stock scb = new Stock("FIFO");  
    scb.buy(5, 100);  
    scb.sell(10, 150);  
    scb.buy(5, 150);  
    scb.sell(10, 150);  
}
```

Output

Sell command rejected

Realized P/L = 250.0 Unrealized P/L = 0.0

Java code

```
public static void main(String[] args) {  
    Stock scb = new Stock("LIFO");  
    scb.buy(5, 100);  
    scb.sell(10, 150);  
    scb.buy(5, 150);  
    scb.sell(0, 125);  
}
```

Output

Sell command rejected

Realized P/L = 0.0 Unrealized P/L = 0.0