Algorithms – Programming Assignment 2

Queues and Randomized Queues

B06602035 李晴妍

E-Mail : b06602035@ntu.edu.tw

Phone : 0987883537

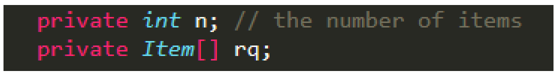
1.

(1) Operating system: Windows
(2) Compiler: IntelliJ IDEA
(3) Text editor / IDE: IntelliJ IDEA

2. Explain briefly how you implemented the randomized queue and deque. Which data structure did you choose (array, linked list, etc.) and why?

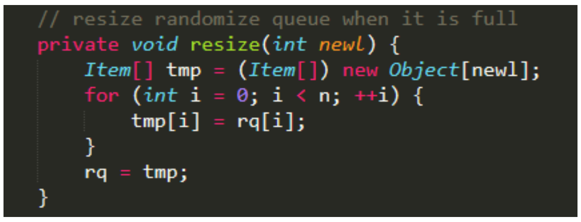
(1) Randomized Queue

我用 array 去實現 Randomized Queue。我認為相比起 linked list，array 雖然在 size 上比較難調整，需要比較多的空間，但在 enqueue 與 dequeue 時可以非常簡單的實施。同時多設立一個變數 n，紀錄現在 randomized queue 中的 item 數量。



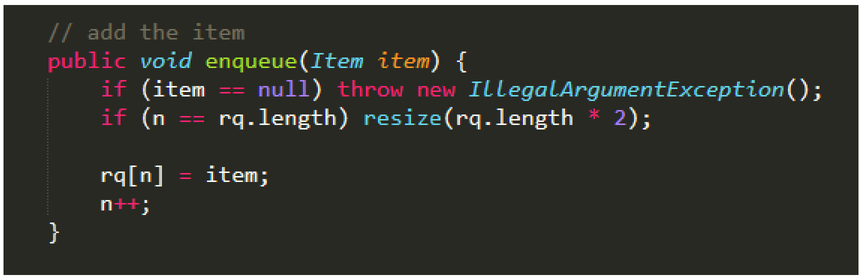
```java
private int n; // the number of items
private Item[] rq;
```

resize 的部分是用之前老師說的方法，當 item 填滿 array 的時候，double 原本的 size。



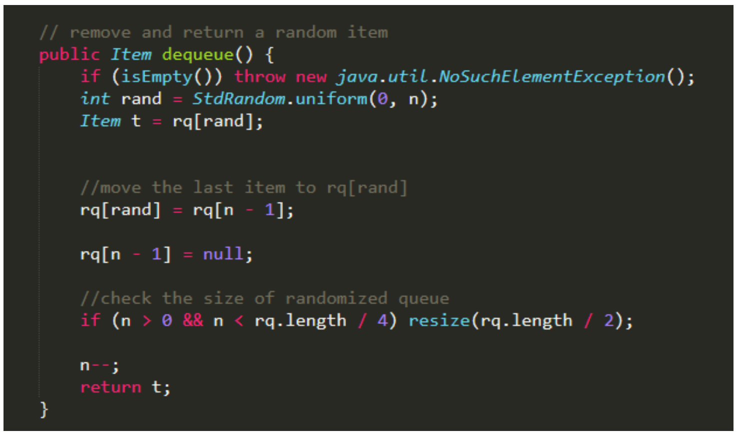
```java
// resize randomize queue when it is full
private void resize(int newl) {
    Item[] tmp = (Item[]) new Object[newl];
    for (int i = 0; i < n; ++i) {
        tmp[i] = rq[i];
    }
    rq = tmp;
}
```

enqueue: 當 queue 還有足夠的 memory 時，將 input item 放入 queue 的最後面。同時檢查當前 queue 的大小。



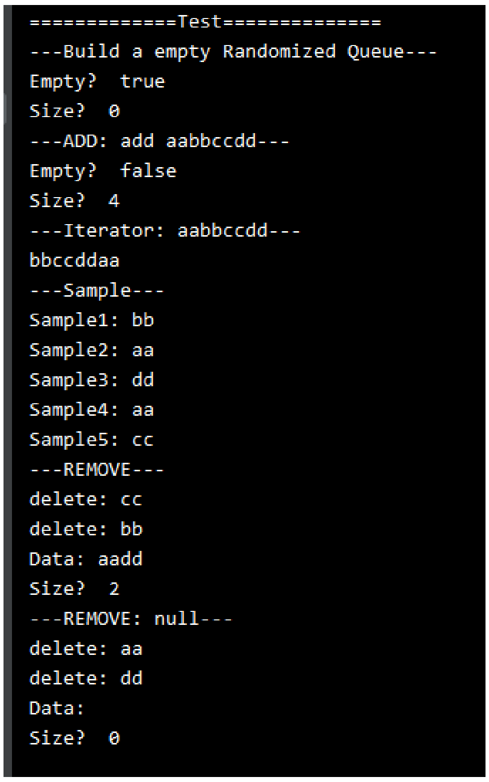
```java
// add the item
public void enqueue(Item item) {
    if (item == null) throw new IllegalArgumentException();
    if (n == rq.length) resize(rq.length * 2);

    rq[n] = item;
    n++;
}
```

dequeue: 隨機取出 queue 中的 item。實作方式是透過 Std.Random 去產生 random index。取出資料後將 queue 最後一個 data 放入 random index。



```java
// remove and return a random item
public Item dequeue() {
    if (isEmpty()) throw new java.util.NoSuchElementException();
    int rand = StdRandom.uniform(0, n);
    Item t = rq[rand];


    //move the last item to rq[rand]
    rq[rand] = rq[n - 1];

    rq[n - 1] = null;

    //check the size of randomized queue
    if (n > 0 && n < rq.length / 4) resize(rq.length / 2);

    n--;
    return t;
}
```

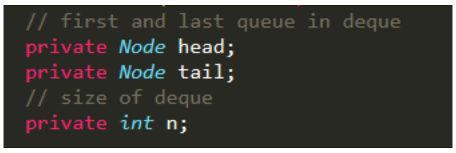
Test:



```
=============Test==============
---Build a empty Randomized Queue---
Empty?  true
Size?  0
---ADD: add aabbccdd---
Empty?  false
Size?  4
---Iterator: aabbccdd---
bbccddaa
---Sample---
Sample1: bb
Sample2: aa
Sample3: dd
Sample4: aa
Sample5: cc
---REMOVE---
delete: cc
delete: bb
Data: aadd
Size?  2
---REMOVE: null---
delete: aa
delete: dd
Data:
Size?  0
```

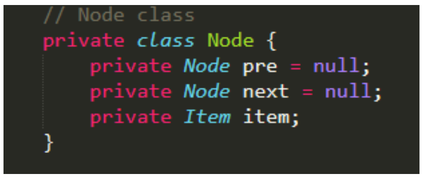
(2) Deque

使用 linked list 實作。設變數 head 及 tail 記住 deque 的 first 和 last node。同時設立變數 n 紀錄 deque 的大小。



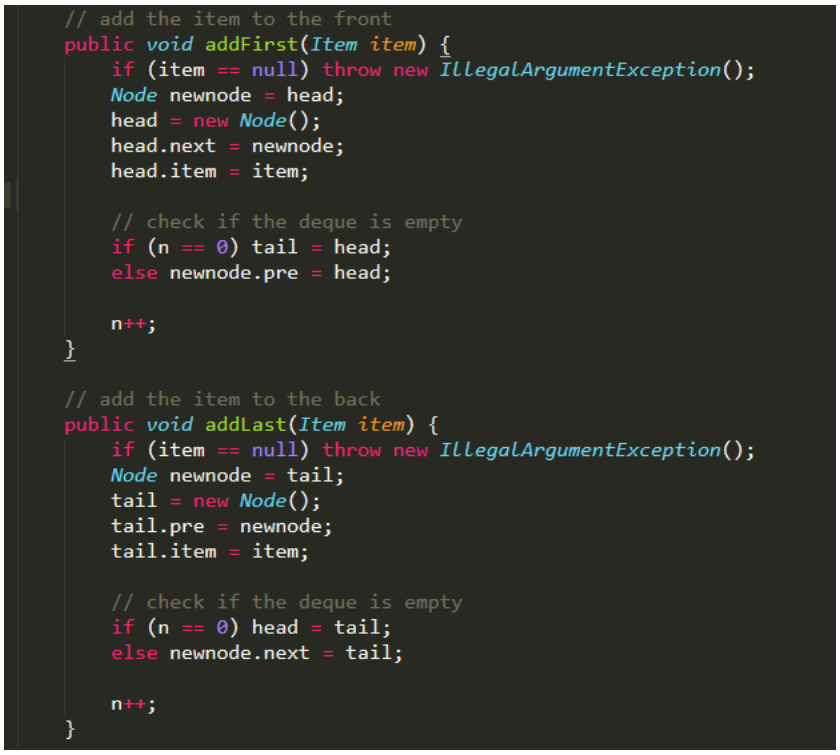
```java
// first and last queue in deque
private Node head;
private Node tail;
// size of deque
private int n;
```

因為 java 沒有 pointer，所以建立一個 private class Node，包含 previous node、next node 及資料 Item。



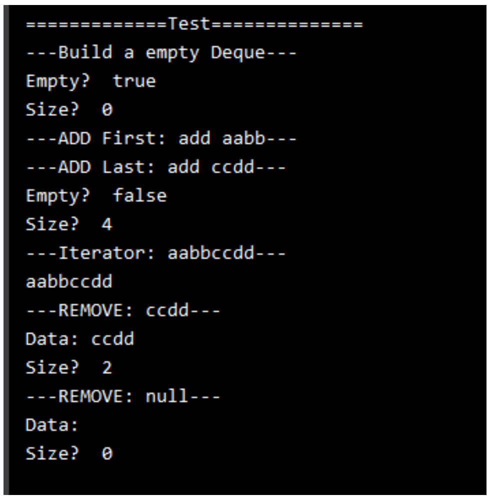
```java
    // Node class
    private class Node {
        private Node pre = null;
        private Node next = null;
        private Item item;
    }
```

addFirst / addLast: 根據 deque 是否 empty 去對 head 及 tail 做不同的更改。Remove 的部分也是相同方式。



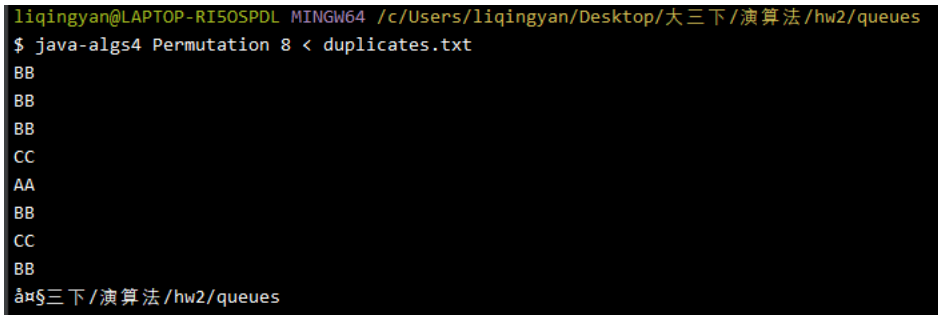
```java
// add the item to the front
public void addFirst(Item item) {
    if (item == null) throw new IllegalArgumentException();
    Node newnode = head;
    head = new Node();
    head.next = newnode;
    head.item = item;

    // check if the deque is empty
    if (n == 0) tail = head;
    else newnode.pre = head;

    n++;
}

// add the item to the back
public void addLast(Item item) {
    if (item == null) throw new IllegalArgumentException();
    Node newnode = tail;
    tail = new Node();
    tail.pre = newnode;
    tail.item = item;

    // check if the deque is empty
    if (n == 0) head = tail;
    else newnode.next = tail;

    n++;
}
```

Test:

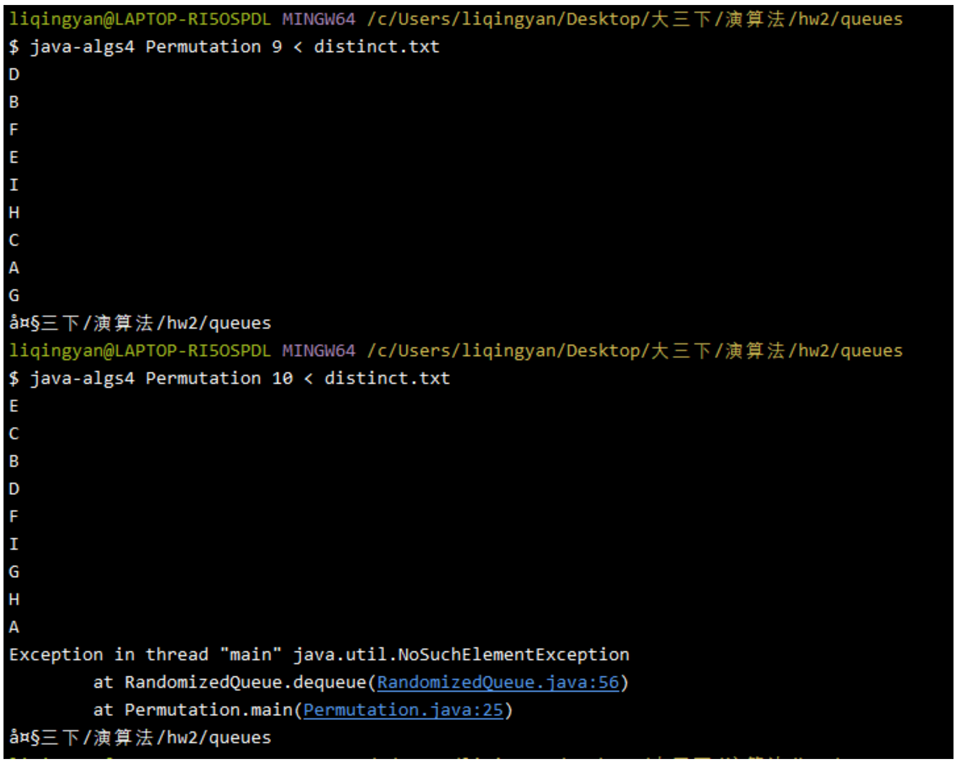


```
=============Test==============
---Build a empty Deque---
Empty?  true
Size?  0
---ADD First: add aabb---
---ADD Last: add ccdd---
Empty?  false
Size?  4
---Iterator: aabbccdd---
aabbccdd
---REMOVE: ccdd---
Data: ccdd
Size?  2
---REMOVE: null---
Data:
Size?  0
```

(3) Permulation



```
liqingyan@LAPTOP-RI5OSPDL MINGW64 /c/Users/liqingyan/Desktop/大三下/演算法/hw2/queues
$ java-algs4 Permutation 8 < duplicates.txt
BB
BB
BB
CC
AA
BB
CC
BB
å¤§三下/演算法/hw2/queues
```



```
liqingyan@LAPTOP-RI5OSPDL MINGW64 /c/Users/liqingyan/Desktop/大三下/演算法/hw2/queues
$ java-algs4 Permutation 9 < distinct.txt
D
B
F
E
I
H
C
A
G
å¤§三下/演算法/hw2/queues
liqingyan@LAPTOP-RI5OSPDL MINGW64 /c/Users/liqingyan/Desktop/大三下/演算法/hw2/queues
$ java-algs4 Permutation 10 < distinct.txt
E
C
B
D
F
I
G
H
A
Exception in thread "main" java.util.NoSuchElementException
        at RandomizedQueue.dequeue(RandomizedQueue.java:56)
        at Permutation.main(Permutation.java:25)
å¤§三下/演算法/hw2/queues
```

3. How much memory (in bytes) do your data types use to store n items in the worst case? Use the 64-bit memory cost model from Section 1.4 of the textbook and use tilde notation to simplify your answer. Briefly justify your answers and show your work.
   Do not include the memory for the items themselves (as this memory is allocated by the client and depends on the item type) or for any iterators, but do include the memory for the references to the items (in the underlying array or linked list).

   Randomized Queue: ~ 32.00 n + 88.00 (R^2 = 1.000) bytes

   Total memory usage after inserting 4096 items, then successively deleting items, seeking values of n where memory usage is maximized as a function of n.

```
                   n           bytes
---------------------------------------------------------
=> passed       2047          65592
=> passed       1023          32824
=> passed        511          16440
=> passed        255           8248
=> passed        127           4152
=> passed         63           2104
=> passed         31           1080
=> passed         15            568
==> 8/8 tests passed
```

   Deque: ~ 48.00 n + 40.00 (R^2 = 1.000) bytes

   Total memory usage after inserting n items, where n is a power of 2.

```
                   n           bytes
---------------------------------------------------------
=> passed         32           1576
=> passed         64           3112
=> passed        128           6184
=> passed        256          12328
=> passed        512          24616
=> passed       1024          49192
=> passed       2048          98344
=> passed       4096         196648
=> passed       8192         393256
==> 9/9 tests passed
```

4. Describe whatever help (if any) that you received. Don't include readings, lectures, and precepts, but do include any help from people (including course staff, lab TAs, classmates, and friends) and attribute them by name.

   https://github.com/Zhanlgu/Princeton-Algorithms-assignment-2-Deques-and-Randomized-Queues/blob/master/RandomizedQueue.java

5. Describe any serious problems you encountered.

   就是沒有 pointer 這件事，寫起來有點怪怪的，在使用記憶體的部分要更小心一點。

6. List any other comments here. Feel free to provide any feedback on how much you learned from doing the assignment, and whether you enjoyed doing it.

   我之前比較習慣寫 C++，這次換到 java 是一次蠻有趣的嘗試 ~ 不能用 pointer，就要改用其他方式完成，而且 IntelliJ IDEA 也很好用，很棒！