Homework 6: Priority Queue and Heap

CHÚ Ý:

1. Lựa chọn các gói bài tập để thực hiện:

   Combo 1: Bài 1, Bài 2, Bài 3 (cơ bản)

   Combo 2: Bài 1, Bài 4, Bài 5 (nâng cao 1)

   Combo 2: Bài 2, Bài 3, Bài 6 (nâng cao 2)

   Combo 4: Bài 5, Bài 6 (nâng cao 3)

2. Trong bài nộp có file .doc hoặc .txt thuyết minh về gói bài tập thực hiện và những nội dung cần giải thích về bài làm: thuật toán được chọn, tài liệu tham khảo, định dạng input, yêu cầu hệ thống...

<PriorityQueue>

Bài 1. Tạo giao diện các phần tử và hàng đợi ưu tiên PriotityQueueInterface như sau:

```
public interface Entry <K,E> {
     K getKey();    //K là khóa của phần tử
     E getValue();  //E là giá trị phần tử
}
public interface PriorityQueueInterface<K , E> {
     public int size();
     public boolean isEmpty();
     public void insert(Entry<K, E> entry);  //thêm một entry vào PQ
     public void insert(K k, E e);  //thêm phần tử có key k và giá trị e vào PQ
     public Entry<k, E> removeMin();     //loại phần tử có giá trị nhỏ nhất
     public Entry<k, E> min();           //trả về phần tử có key nhỏ nhất
}
```

1.1 Xây dựng kiểu dữ liệu UnsortedArrayPriorityQueue sử dụng mảng, cài đặt giao diện PriorityQueueInterface đã xây dựng ở trên với lược đồ gợi ý như sau:

```
public class UnsortedArrayPriorityQueue<K extends Comparable, E> implements
                                                        PriorityQueueInterface {
     protected class ArrEntry<K, E> implements Entry<K, E>{
          K key;
          E element;
          public ArrEntry (K k, E e){

          }
     }
     ArrEntry<K, E> [] array;
     int n = 0;
     int defaultsize = 1000;
}
```

1.2 Xây dựng kiểu dữ liệu SortedArrayPriorityQueue sử dụng mảng, cài đặt giao diện PriorityQueueInterface đã xây dựng ở trên với lược đồ gợi ý như trong bài 1.1

```
public class SortedArrayPriorityQueue<K extends Comparable, E> implements
                                                        PriorityQueueInterface {
}
```

1.3 Xây dựng kiểu dữ liệu UnsortedLinkedPriorityQueue sử dụng danh sách liên kết, cài đặt giao diện PriorityQueueInterface đã xây dựng ở trên với lược đồ gợi ý như sau:

```
public class UnsortedLinkedPriorityQueue<K extends Comparable, E> implements
                                                        PriorityQueueInterface {
    protected class NodeEntry<K, E> implements Entry<K, E>{
        private K key;
        private E element;
        private NodeEntry<K, E> next;
        public ArrEntry (K k, E e){

        }
    }
    private NodeEntry<K,E> head;
    private NodeEntry<K,E> tail;
    int n = 0;
}
```

1.4 Xây dựng kiểu dữ liệu SortedLinkPriorityQueue sử dụng danh sách liên kết, cài đặt giao diện PriorityQueueInterface đã xây dựng ở trên với lược đồ gợi ý như trong bài 1.3:

```
public class SortedLinkedPriorityQueue<K extends Comparable, E> implements
                                                        PriorityQueueInterface {
}
```

1.5 Viết hàm test các kiểu dữ liệu PriorityQueue đã triển khai với:

- Danh sách các số nguyên, giá trị phần tử dùng làm khóa.

- Danh sách các đối tượng có khóa và giá trị khác nhau. Ví dụ: đối tượng *hàng hóa* bao gồm *tên hàng hóa* (giá trị), *giá tiền* (khóa).

<Heap>

Bài 2. Xây dựng cấu trúc dữ liệu hàng đợi ưu tiên sử dụng heap cài đặt bằng mảng với lược đồ như sau:

```
public class MinHeapPriorityQueue<K extends Comparable, E> extends
                                                    SortedArrayPriorityQueue {
    ArrEntry<K,E> heapPQ[];

    //Các phương thức bổ sung
    protected void upHeap()     //vun lên
    protected void downHeap()   //vun xuống
}
```

Bài 3. Sử dụng cấu trúc dữ liệu HeapPriorityQueue để viết hàm HeapSort sắp xếp dãy số theo thứ tự tăng dần.

Chạy và so sánh thời gian thực hiện của 4 thuật toán sắp xếp SelectionSort, HeapSort, QuickSort và MergeSort.

PHẦN BÀI TẬP NÂNG CAO VÀ ỨNG DỤNG

Bài 4. Sử dụng 4 kiểu dữ liệu PriorityQueue đã xây dựng ở các bài 1.2-1.4 để thực hiện các yêu cầu sau:

- Lập danh sách *n* đối tượng (*d*,*n*). Để đơn giản đối tượng *d* có giá trị nguyên và sử dụng giá trị làm khóa.

- Thực hiện các thao tác thêm phần tử vào danh sách (insert), lấy ra phần tử nhỏ nhất (removeMin) với các danh sách có độ dài *n* khác nhau, với các kiểu PriorityQueue đã được cài đặt. Lập bảng so sánh thời gian thực hiện (milisecond) các thao tác có dạng như sau:

Methods & PriorityQueue \ n		10^3	10^4	10^5	10^6	10^7	10^8
insert	UnsortedArray						
	SortedArray						
	UnsortedLinked						
	SortedLinked						
removeMin	UnsortedArray						
	SortedArray						
	UnsortedLinked						
	SortedLinked						

Bài 5. Sử dụng kiểu PriorityQueue đã xây dựng ở bài 1 viết chương trình mô phỏng hệ thống điều hành không lưu đơn giản, quản lí các sự kiện máy bay cất cánh và hạ cánh tại sân bay. (*Xem bài tập R-9.4 sách M.Goodrich, trang 395)*

Hệ thống được mô tả như sau: Mỗi một máy bay đăng kí sự kiện cất cánh và hạ cánh với một nhãn thời gian, là thời gian mà sự kiện đó dự kiến sẽ xảy ra. Chương trình mô phỏng cần thực hiện một cách hiệu quả 2 chức năng sau:

- Thêm vào hệ thống một đăng kí sự kiện cất/hạ cánh, là sự kiện sẽ xảy ra với một *nhãn thời gian* xác định.
- Lấy ra sự kiện cất/hạ cánh chuẩn bị cho công tác điều hành, là sự kiện sắp xảy ra với *nhãn thời gian* nhỏ nhất.

Bài 6. Sử dụng HeapPriorityQueue ở bài tập 2 viết chương trình mô phỏng hệ thống điều khiển giao dịch chứng khoán đơn giản, quản lí các lệnh mua và bán. (*Xem bài tập C-9.48, P-9.54 sách M.Goodrich, trang 395).*