2019년 알고리즘

- HW 05 -

제출일자	2019.10.09.
이 름	장수훈
학 번	201402414
분 반	01

Priority Queue

```
public static void main(String[] args) throws IOException {
    FileReader fr = new FileReader("C:\\Users\\micke\\eclipse-workspace\\ALG\\src\\week05\\data05.txt");
    BufferedReader br = new BufferedReader(fr);
    String line = br.readLine();

    int key;
    String value;

    ArrayList<Integer> list = new ArrayList<Integer>();
    int index = 0;
    //Priority heap = new Priority(list, list.size());
    Hashtable<Integer, String> table = new Hashtable<Integer, String>();

    try {
        while (line != null) {
            StringTokenizer st = new StringTokenizer(line, ",");

            key = Integer.parseInt(st.nextToken());
            value = st.nextToken();
            table.put(key, value);
            list.add(key);
            line = br.readLine();
            index++;
        }

        build_max_heap(list, list.size());
        Scanner input = new Scanner(System.in);
        Scanner input1 = new Scanner(System.in);
        int input_key;
        String input_value;
        int number;
```

코드 설명

text파일을 읽어와 숫자는 key값, 과목명은 value값으로 해시테이블에 넣었다. 동시에 동적 할당을 하기 위해 arraylist를 이용해 key값만 add해주었다. 이후 입력받는 값에 따라 동작을 다르게 한다.

```java
public static void max_heapify(ArrayList<Integer> list, int i) {
    int largest = i;
    int L = left_child(i);
    int R = right_child(i);
    int heap_size = list.size();

    if ((L < heap_size) && (list.get(L) > list.get(i))) {
        largest = L;
    } else {
        largest = i;
    }
    if ((R < heap_size) && (list.get(R) > list.get(largest))) {
        largest = R;
    }
    if (largest != i) {
        Collections.swap(list, i, largest);
        max_heapify(list, largest);
    }
}

public static int parent(int i) {
    return i / 2;
}

public static int left_child(int i) {
    return 2 * i + 1;
}

public static int right_child(int i) {
    return 2 * i + 2;
}

public static void build_max_heap(ArrayList<Integer> list, int size) {
    for (int i = size / 2; i >= 0; i--) {
        max_heapify(list, i);
    }
}
```

코드 설명

max_heapify 함수는 수도코드대로 구현을 하였다. 트리를 비교해 max힙이 아닐 경우 교체하는 작업이다. build_max_heap은 입력 받은 리스트를 최대 힙으로 만드는 함수다. 자식을 갖는 마지막 노드부터 루트 노드까지 순서대로 검사하여 max_heapity함수를 실행하였다.

```java
public static void increase_key(ArrayList<Integer> S, int x, int k) {
    int count = 0;
    for (int i = 0; i < S.size(); i++) {
        if (S.get(i) == x) {
            break;
        }
        count++;
    }
    S.set(count, k);
    build_max_heap(S, S.size());
}

public static void insert(ArrayList<Integer> S, int x) {
    S.add(x);
    build_max_heap(S, S.size());
}

public static int max(ArrayList<Integer> S) {
    return S.get(0);
}

public static void extract_max(ArrayList<Integer> S) {
    Collections.swap(S, 0, S.size() - 1);
    S.remove(S.size() - 1);
    build_max_heap(S, S.size());
}

public static void delete(ArrayList<Integer> S, int x) {
    int count = 0;
    for (int i = 0; i < S.size(); i++) {
        if (S.get(i) == x) {
            break;
        }
        count++;
    }
    S.remove(count);
    build_max_heap(S, S.size());
}
```

코드 설명

increase_key함수는 원소 x의 킷값을 k로 증가시킨다. 이때 k는 x의 현재 킷값보다 작아지는 걸 방지하기 위해 main함수에서 x > k인 상황을 제외시켰다. 킷값 x를 찾기 위해 count변수를 사용해 x와 같아질 때 까지 증가시켜서 index를 구해서 set함수를 통해 변경할 k로 킷값을 변경해주었다.

insert 함수는 입력받은 x값을 key값으로 arraylist에 추가시킨 후 히피화 해주었다.

max함수는 리스트의 최대값인 첫 번째index를 반환해 주었다.

extract_max 함수는 ppt에 나와 있는 대로 첫 번쨔 인덱스와 마지막 인덱스를 교체 한 후 마지막 인덱스인 최대값을 삭제 후 히피화 해주었다.

delete함수는 x값을 킷값으로 가지는 index를 count로 구해준 뒤 지워준 후 히피화 하였다.

결과값

```
**** 현재 우선 순위 큐에 저장되어 있는 작업 대기 목록은 다음과 같습니다. ****

 230, 수치해석
  70, 자료구조 및 실습
 150, 파일처리론
  40, 컴퓨터 구조2
  60, 객체지향설계
  80, 기초물리학
  98, 계산이론
  38, 논리회로 및 실험
  30, 컴퓨터 구조1
  41, 고급프로그램설계
  45, 소프트웨어 설계
  56, 소프트웨어 공학
   9, 선형대수
   1, 컴퓨터프로그래밍1
   3, 컴퓨터프로그래밍2
  27, 이산수학
  29, 프로그래밍 언어

----------------------------------------
 1. 작업 추가  2. 최대값 3.최대 우선순위 작업 처리
 4. 원소 키값 증가         5. 작업 제거     6.종료
----------------------------------------
```

```
추가할 키값 :
22
작업명 :
테니스
**** 현재 우선 순위 큐에 저장되어 있는 작업 대기 목록은 다음과 같습니다. ****

 230, 수치해석
  70, 자료구조 및 실습
 150, 파일처리론
  40, 컴퓨터 구조2
  60, 객체지향설계
  80, 기초물리학
  98, 계산이론
  38, 논리회로 및 실험
  30, 컴퓨터 구조1
  41, 고급프로그램설계
  45, 소프트웨어 설계
  56, 소프트웨어 공학
   9, 선형대수
   1, 컴퓨터프로그래밍1
   3, 컴퓨터프로그래밍2
  27, 이산수학
  29, 프로그래밍 언어
  22, 테니스

----------------------------------------
 1. 작업 추가  2. 최대값 3.최대 우선순위 작업 처리
 4. 원소 키값 증가         5. 작업 제거     6.종료
----------------------------------------
```

```
----------------------------------------
 1. 작업 추가  2. 최대값 3.최대 우선순위 작업 처리
 4. 원소 키값 증가         5. 작업 제거     6.종료
----------------------------------------
2
230, 수치해석
```

```
3
**** 현재 우선 순위 큐에 저장되어 있는 작업 대기 목록은 다음과 같습니다. ****

 150, 파일처리론
  70, 자료구조 및 실습
  98, 계산이론
  40, 컴퓨터 구조2
  60, 객체지향설계
  80, 기초물리학
  22, 테니스
  38, 논리회로 및 실험
  30, 컴퓨터 구조1
  41, 고급프로그램설계
  45, 소프트웨어 설계
  56, 소프트웨어 공학
   9, 선형대수
   1, 컴퓨터프로그래밍1
   3, 컴퓨터프로그래밍2
  27, 이산수학
  29, 프로그래밍 언어
```

```
----------------------------------------
 1. 작업 추가  2. 최대값 3.최대 우선순위 작업 처리
 4. 원소 키값 증가         5. 작업 제거     6.종료
----------------------------------------
4
변경할 키값 :
1
증가한 키값 :
2

**** 현재 우선 순위 큐에 저장되어 있는 작업 대기 목록은 다음과 같습니다. ****

 150, 파일처리론
  70, 자료구조 및 실습
  98, 계산이론
  40, 컴퓨터 구조2
  60, 객체지향설계
  80, 기초물리학
  22, 테니스
  38, 논리회로 및 실험
  30, 컴퓨터 구조1
  41, 고급프로그램설계
  45, 소프트웨어 설계
  56, 소프트웨어 공학
   9, 선형대수
   2, 컴퓨터프로그래밍1
   3, 컴퓨터프로그래밍2
  27, 이산수학
  29, 프로그래밍 언어
```

```
----------------------------------------
 1. 작업 추가  2. 최대값 3.최대 우선순위 작업 처리
 4. 원소 키값 증가         5. 작업 제거     6.종료
----------------------------------------
5
제거할 키값 :
2
컴퓨터프로그래밍1(이)가 제거되었습니다.

**** 현재 우선 순위 큐에 저장되어 있는 작업 대기 목록은 다음과 같습니다. ****

 150, 파일처리론
  70, 자료구조 및 실습
  98, 계산이론
  40, 컴퓨터 구조2
  60, 객체지향설계
  80, 기초물리학
  27, 이산수학
  38, 논리회로 및 실험
  30, 컴퓨터 구조1
  41, 고급프로그램설계
  45, 소프트웨어 설계
  56, 소프트웨어 공학
   9, 선형대수
   3, 컴퓨터프로그래밍2
  22, 테니스
  29, 프로그래밍 언어
```