Homework 4

자료구조. 2024 1학기. 총 5 문제. 제출일: 5/28

1. 키(key) 값이 다음과 같은 자료들을 이진탐색트리(BST, Binary Search Tree)에 저장한다고 하자.

   9, 14, 4, 6, 15, 2, 19, 8, 3, 16, 20, 5, 1, 18, 7, 10, 17, 13, 11, 12

   1) 위 순서로 자료가 삽입되었을 때, 결과로 만들어지는 이진탐색트리를 보이시오.

   2) 위의 트리를 in-order 순회(traverse)하였을 때 방문하는 노드의 순서를 쓰시오.

   3) 위 트리에서 14를 삭제한 이후의 BST 모습을 나타내시오. 삭제되는 노드를 대체하는 노드는 left subtree에서 찾는 것으로 한다.

2. 정수 id와 priority(1, 2, 3)를 가진 process들이 CPU에서 수행되기 위해 다음과 같이 process_queue 라는 배열에 우선순위(priority)를 key로 하여 최대 히프(max heap)로 저장되어 있다고 하자.

```
typedef struct{
        int    key;   /* priority */
        int    id;
} element;
element heap[MAX_ELEMENTS];
```

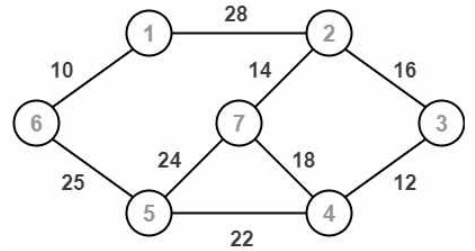
process_queue =

	1	2	3	4	5	6	7	8	9	10
key	3	2	3	2	1	2	1	1	2	1
id	009	014	004	006	015	002	019	008	003	016

   1) 위의 히프를 트리 모양으로 나타내 보시오.

   2) 위 히프에서 delete max를 3번 수행할 때 반환되는 process id를 순서대로 쓰고, 결과 process_queue의 내용을 보이시오.

3. 다음의 무방향 가중치 그래프(undirected weighted graph)를 보고 답하시오



1) 이 그래프를 인접리스트(adjacency list)로 표현해보시오. 인접한 정점이 여러개일 때, 연결되는 정점의 순서는 정점 번호의 오름차순으로 하시오.

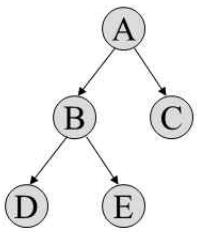
2) 위의 그래프를 7번 정점에서부터 너비우선 탐색(Breadth-First Search)하여 얻어지는 신장트리 (spanning tree)를 보이시오.

3) Kruskal의 알고리즘을 적용하여 위 그래프의 최소신장트리(MST, Minimum Spanning Tree)를 찾는 과정을 보이시오.

4. (Programming) Binary Tree

이진트리(binary tree)에서 root 노드로부터 각 leaf 노드까지의 path를 모두 출력하는 print_paths 함수를 작성하시오. 이진트리는 linked representation으로 구현한다.
(Hint: stack을 이용하고, 재귀적인 함수 형태로 작성하시오)

ex>



→

A, B, D
A, B, E
A, C

```
typedef struct TreeNode {
        int             data;
        TreeNode        *left;
        TreeNode        *right;
} TreeNode;

void print_paths(TreeNode *root);
```

5. (Programming) Graph

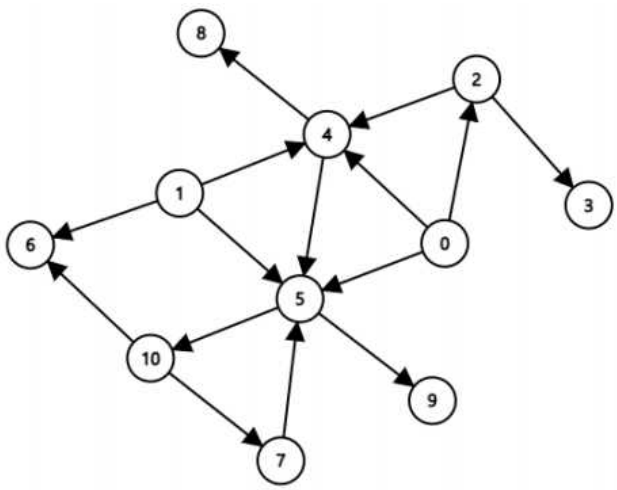
Directed graph에서 특정 vertex와 가까운 vertex들을 찾는 알고리즘을 구현하고자 한다.

1) Breadth-First Search (BFS) 알고리즘을 활용하여 ID가 v인 vertex로부터 distance가 k보다 작거나 같은 vertex들의 리스트를 ID 오름차순으로 출력하는 함수 get_close_vertices(int v, int k) 를 작성하시오.

2) 다음과 같은 두 가지 graph를 대상으로 테스트한 결과를 보이시오. 테스트 프로그램은 그래프의 edge들이 나열된 텍스트 파일을 읽어 adjacency list를 만들고, get_close_vertices(v, k)를 호출한다.

a. test.txt

0,2
0,4
0,5
1,4
1,5
2,3
2,4
4,5
4,8
5,10
10,7
10,6
5,9
7,5
1,6



get_close_vertices(1, 2) → 4, 5, 6, 8, 9, 10

b. facebook.txt (첨부파일)
: Facebook Large Page-Page Network 데이터셋. 22,470 vertices, 171,002 edges

get_close_vertices(1234, 3) → ?