

# Data Structure

Practical Class - Week 10

# Exercise01 – Skeleton 1

---

```
1  #pragma warning(disable: 4996)
2
3  #include<stdio.h>
4  #include<stdbool.h>
5  #include<stdlib.h>
6  #include<memory.h>
7  #include<malloc.h>
8
9  typedef char element;
10
11 struct node;
12 struct tree;
13 typedef struct node* node_ptr;
14 typedef struct tree* tree_ptr;
15
16 typedef struct node {
17     element key;          // data field
18     node_ptr left;        // pointer of the left child node
19     node_ptr right;       // pointer of the right child node
20     node_ptr parent;      // pointer of the parent node
21 } node;
22
23 typedef struct tree{
24     node_ptr root;        // pointer of the root node of tree.
25 };
26
27 tree_ptr make_tree(){
28     tree_ptr new_tree = (tree_ptr)malloc(sizeof(struct tree));
29     new_tree->root = NULL;
30     return new_tree;
31 }
```

## Exercise01 - Skeleton 2

---

```
33  /**
34   * @brief      해당 노드의 서브 트리를 in-order 순서로 출력해주는 함수
35   *              올바른 BST의 경우 in-order로 출력하면 오름차순으로 정렬되어 출력된다.
36   *              [TIP] 테스트용으로 중간 중간 사용하면 유용하다.
37   *
38   * @param node  탐색과 출력을 수행 할 노드의 포인터
39   */
40  void print_inorder(node_ptr node){
41      if (node->parent == NULL){
42          printf("{ ");
43      }
44
45      if (node != NULL){
46          if (node->left != NULL){
47              print_inorder(node->left);
48              printf(", ");
49          }
50
51          printf("%c", node->key);
52
53          if (node->right != NULL){
54              printf(", ");
55              print_inorder(node->right);
56          }
57      }
58
59      if (node->parent == NULL){
60          printf(" }\n");
61      }
62  }
```

# Exercise01 – Problem 1

---

```
64  /**
65   * @brief 새로운 key값을 가진 노드를 생성한 후 BST에 삽입한다.
66   *        만약, 중복된 key가 이미 존재한다면 에러 메시지를 출력하고 NULL을 반환한다.
67   *
68   * @param tree        노드를 추가 할 트리의 포인터
69   * @param new_key      새로 추가 할 노드의 key 값
70   * @return node_ptr    새로 생성된 노드의 포인터
71   */
72  node_ptr insert(tree_ptr tree, element new_key){
73      node_ptr parent = NULL;
74      node_ptr candidate = tree->root;
75
76      node_ptr new_node = (node_ptr)malloc(sizeof(struct node));
77      new_node->left = NULL;
78      new_node->right = NULL;
79      new_node->parent = NULL;
80      new_node->key = new_key;
81
82      // tree의 루트가 없는 경우 이 노드를 루트로 저장한 후 종료.
83      if (tree->root == NULL){
84          tree->root = new_node;
85          return new_node;
86      }
87
88
89
90
91
92      return new_node;
93  }
```

Problem 1

## Exercise01 – Problem 2

---

```
95  /**
96   * @brief   BST에서 target_key값을 가진 노드를 찾아서 포인터를 반환한다.
97   *          만약, 그런 노드가 존재하지 않는다면 에러 메시지를 출력한 후 NULL을 반환한다.
98   *
99   * @param tree      검색을 수행 할 트리의 포인터
100  * @param target_key  검색 할 key값
101  * @return node_ptr   해당 key값을 가지는 노드의 포인터
102  */
103  node_ptr find(tree_ptr tree, element target_key){
104      node_ptr parent = NULL;
105      node_ptr candidate = tree->root;
106
107
108
109      Problem 2
110
111
112
113      return candidate;
114  }
```

# Exercise01 - Problem 3, 4

```
116  /**
117   * @brief  BST에서 target_key값을 가진 노드를 찾아서 삭제 후 할당 해제한다.
118   *        만약, 그런 노드가 존재하지 않는다면 아무 행동도 하지 않는다.
119   *
120   * @param tree      검색을 수행 할 트리의 포인터
121   * @param target_key 검색 할 key값
122   * @return node_ptr  해당 key값을 가지는 노드의 포인터
123   */
124  void delete_node(tree_ptr tree, element target_key){
125      node_ptr parent;
126      node_ptr candidate;
127
128      // 해당 key값을 가진 노드가 존재하는지 검사해 본다.
129      candidate = find(tree, target_key);
130
131      if (candidate == NULL){
132          // 해당 key값을 가진 노드가 존재하지 않는 경우
133          return;
134      }
135
136      // 해당 key값을 가진 노드 candidate가 존재하는 경우
137      parent = candidate->parent;
138
139      if (candidate->left == NULL || candidate->right == NULL){
140          /**
141           * CASE 1 : candidate가 하나의 자식을 가지거나 자식을 가지지 않는 경우.
142           */
143
144
145
146
147          return;
148      }
149      else{
150          /**
151           * CASE 2 : candidate가 정확히 두 개의 자식 노드를 가지는 경우
152           */
153
154          node_ptr predecessor = candidate->left;
155          node_ptr parent_of_predecessor = candidate;
156
157
158
159
160          return;
161      }
162  }
```

<HINT 1>

자식이 하나 인 경우, 삭제 될 노드의 자리를  
기존의 자식 노드로 대체하면 된다

<HINT 2>

두 개의 자식을 가진다

⇔ Predecessor는 항상 candidate의 자손이다.

Problem 3

Problem 4

# Exercise01 - Example

---

테스트를 위한 코드는 아래 URL에서 Copy & Paste 하세요.

- <https://gist.github.com/waps12b/93ec2a8d83cccc83cb5501ea0c730786>

test\_case\_1() 함수의 실행 예시

- 보고서에는 test\_case\_2() 함수를 실행 한 결과를 캡처해서 첨부하세요!

# Exercise02 – Skeleton 1

---

```
1  #pragma warning(disable: 4996)
2  #include<stdio.h>
3  #include<stdbool.h>
4  #include<memory.h>
5  #include<malloc.h>
6  #include<stdbool.h>
7
8  typedef struct disjoint_set{
9      int size;          // number of nodes in the set
10     int* parents;      // parents[i] := the boss's index of the group that node #i belongs to.
11 } disjoint_set;
12
13
14 /**
15  * @brief   새로운 집합 정보를 생성한 후 초기화하여 반환하는 함수
16  *
17  * @param number_of_nodes   관리할 노드의 수
18  * @return disjoint_set*   새로 생성된 집합의 포인터
19  */
20 disjoint_set* init_disjoint_set(int number_of_nodes){
21     int node_index;
22     disjoint_set* new_set = (disjoint_set*)malloc(sizeof(disjoint_set));
23     new_set->size = number_of_nodes;
24     new_set->parents = (int*)malloc(sizeof(int)*number_of_nodes);
25
26     // 처음에는 자기 자신을 boss로 가진다.
27     for (node_index = 0; node_index < number_of_nodes; node_index += 1){
28         new_set->parents[node_index] = node_index;
29     }
30     return new_set;
31 }
```



## Exercise02 – Skeleton 2

---

```
33  /**
34   * @brief 할당 해제 해 주는 함수
35   *
36   * @param set_ptr
37   */
38  void release_disjoint_set(disjoint_set* set_ptr){
39      free(set_ptr->parents);
40      free(set_ptr);
41  }
```

# Exercise02 – Problem 1

---

```
43  /**
44   * @brief 노드 u에 대해, 현재 자신이 속한 그룹의 대표 노드(boss)의 인덱스를 반환하는 함수
45   *
46   * @param set_ptr
47   * @param u
48   * @return int
49   */
50  int find_parent(disjoint_set* set_ptr, int u){
51
52
53
54
55
56
57
58
59      //printf("The boss of a group that the node #%d belongs to is node #%d\n", u, u_boss);
60      return u_boss;
61  }
```

Problem 1

## Exercise02 – Problem 2

---

```
63  /**
64   * @brief 노드 u와 v가 속한 두 그룹을 병합하는 함수
65   *
66   * @param set_ptr
67   * @param u
68   * @param v
69   */
70  void union_set(disjoint_set* set_ptr, int u, int v){
71
72
73
74
75
76
77      //printf("Merged two groups of node #%d and node #%d.\n", u, v);
78  }
```

Problem 2

## Exercise02 – Skeleton 3

---

```
80  /**
81   * @brief 두 노드 u와 v가 같은 그룹에 속해 있다면 true, 그렇지 않다면 false를 반환하는 함수
82   *
83   * @param set_ptr
84   * @param u
85   * @param v
86   * @return true
87   * @return false
88   */
89  bool is_same_group(disjoint_set* set_ptr, int u, int v){
90      int u_boss = find_parent(set_ptr, u);
91      int v_boss = find_parent(set_ptr, v);
92      bool yes = (u_boss == v_boss);
93
94      if (yes){
95          printf("Node %d and Node %d are in the same group with the boss %d.\n", u, v, u_boss);
96      }
97      else{
98          printf("Node %d and Node %d are not in the same group.\n", u, v, u_boss);
99      }
100     return yes;
101 }
```

## Exercise02 – Skeleton 4

---

```
103  /**
104   * @brief 현재 disjoint set의 집합 현황을 출력해주는 함수
105   *
106   * @param set_ptr
107   */
108  void print_sets(disjoint_set* set_ptr){
109      int boss = 0;
110      int member = 0;
111      printf("[Set Information]\n");
112      for (boss = 0; boss < set_ptr->size; boss += 1){
113          if (set_ptr->parents[boss] == boss){
114              printf(" (Boss: %d) {" , boss);
115              for (member = 0; member < set_ptr->size; member += 1){
116                  int my_boss = find_parent(set_ptr, member);
117                  if (my_boss == boss){
118                      printf(" %d ", member);
119                  }
120              }
121              printf(" }\n");
122          }
123      }
124      printf("\n");
125  }
```

## Exercise02 - Example

---

테스트를 위한 코드는 아래 URL에서 Copy & Paste 하세요.

- <https://gist.github.com/waps12b/93ec2a8d83cccc83cb5501ea0c730786>

test\_case\_1() 함수의 실행 예시

- 보고서에는 test\_case\_2() 함수를 실행 한 결과를 캡처해서 첨부하세요!

```
C:\Windows\system32\cmd.exe
[Student ID] 201724516
[NAME] HONG KIL-DONG
Node #0 and Node #6 are not in the same group.
Node #6 and Node #0 are not in the same group.
Node #0 and Node #0 are in the same group with the boss #5.
[Set Information]
(Boss: 1) { 1 3 }
(Boss: 4) { 4 }
(Boss: 5) { 0 2 5 6 }
계속하려면 아무 키나 누르십시오 . . .
```