**hw5\_1**

**텍스트, 영수증이(가) 표시된 사진

자동 생성된 설명**

**hw5\_2.c**

1. Code Explanation

1)TreeNode\* tree\_successor(TreeNode\* p)

: we will find the successor of node p

-TreeNode\* y: we will save successor node in y(second case)

-when p's right subtree is not null: find leftmost node of right subtree

-when p's right subtree is null

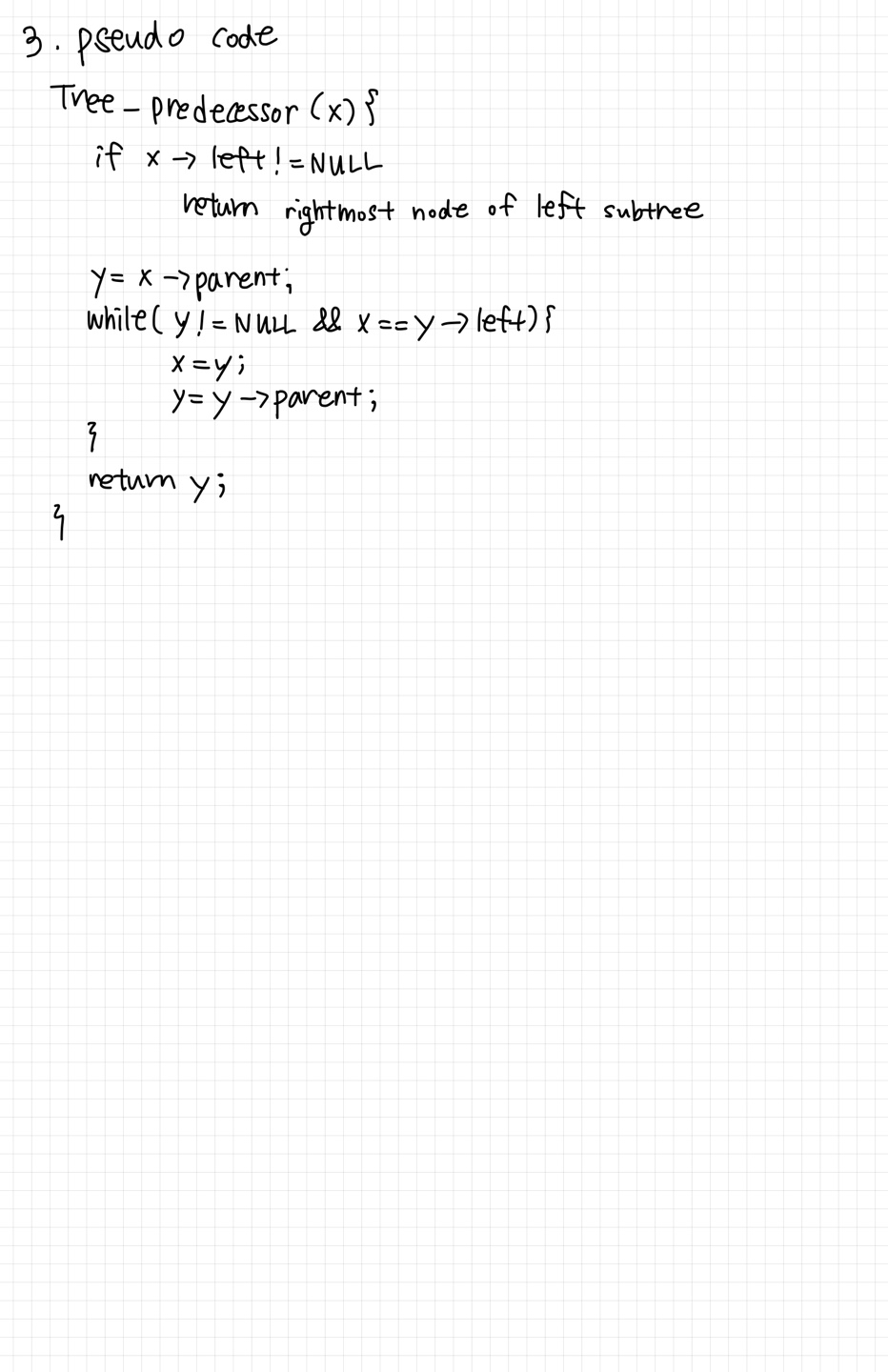
2. Result

텍스트이(가) 표시된 사진

자동 생성된 설명

**hw5\_3.c**

1. Pseudo Code

****

2. Code Explanation

1)TreeNode\* tree\_predecessor(TreeNode\* p)

: we will find the predecessor of node p

-TreeNode\* y: we will save predcessor node in y(second case)

-when p's left subtree is not null: find rightmost node of left subtree

-when p's left subtree is null

2) void main()

while (q->right) q = q->right;// Go to the rightmost node to find the last node of inorder traversal

3. Result

텍스트이(가) 표시된 사진

자동 생성된 설명

**hw5\_4.c**

1. Code Explanation

1) void delete\_node(TreeNode\*\* root, int key)

: Deletion in binary search tree

-TreeNode\* t: node whose data==key

-TreeNode\* p: parent node of t

-TreeNode\* child: child node of t

-TreeNode\* pred: predecessor of node t

-TreeNode\* pred\_p: parent node of pred

//Case3

-Find the predecessor at right subtree

-Keep moving to the right and find the predcessor

-bring pred's right child node to predecessor's original location

-bring predecessor node to the location of node t

2. Result

1) key = 18

텍스트이(가) 표시된 사진

자동 생성된 설명

2) key = 35

텍스트, 모니터, 화면이(가) 표시된 사진

자동 생성된 설명

3) key = 7

텍스트, 모니터, 화면, 검은색이(가) 표시된 사진

자동 생성된 설명