Instrukcja – Algorytmy i struktury danych

Drzewa BST

Za pomocą implementacja wskaźnikowej zaimplementuj drzewo BST. Jako element drzewa możesz przyjąć implementację:

```
class BSTNode
{
public:
    int key;
    BSTNode* Left;
    BSTNode* Right;
    BSTNode* Parent;

    static BSTNode* Insert(BSTNode* node, int key);
    static void printTreePreOrder(BSTNode* node);
    static void printTreePostOrder(BSTNode* node);
    static void printTreeInOrder(BSTNode* node);

    static BSTNode* min(BSTNode* root);
    static BSTNode* max(BSTNode* root);

    static BSTNode* inOrderSuccessor(BSTNode* n);
    static BSTNode* inOrderPredeccessor(BSTNode* n);
};
```

1. Napisz trzy funkcję do przechodzenia drzewa BST i wypisz wynik:
    a. PreOrder,
    b. PostOrder,
    c. InOrder.
2. Napisz funkcję, która znajdzie w strukturze drzewa:
    a. Minimum,
    b. Maksimum.
3. Napisz funkcje, która obliczy wysokość drzewa,
4. Napisz funkcję, która znajdzie:
    a. Następnika podanego węzła,
    b. Poprzednika podanego węzła.
5. Napisz funkcję, która:
    a. Wstawi podany element w odpowiednie miejsce drzewa,
    b. Usunie podany element.