#1:

```cpp
#include <iostream>
#include <queue>
using namespace std;


class Customer {
public:
    string plate;
    string model;
    int priority;

    bool operator<(const Customer& other) const {
        return priority < other.priority;
    }

    friend ostream& operator<<(ostream& os, const Customer& customer) {
        os << customer.plate << " (" << customer.model << ")";
        return os;
    }
};

void insertCustomer(priority_queue<Customer>& custpq, Customer cnew) {
    custpq.push(cnew);
    while (!custpq.empty()) {
        cout << custpq.top() << " ";
        custpq.pop();
    }
    cout << endl;
}


int main() {
    priority_queue<Customer> custpq;
    cout << "Car license plates in order of priority:\n";
    insertCustomer(custpq, { "plate1", "make1", 1 });
    insertCustomer(custpq, { "plate2", "make2", 2 });
    insertCustomer(custpq, { "plate3", "make3", 3 });

    return 0;
}
```

#2:

```cpp
#include <iostream>
using namespace std;

int index(const int arr[], int size);

int main() {
    int arr[] = { 5,4,3,2,1,0 };
    int size = sizeof(arr) / sizeof(arr[0]);

    int smallestIndex = index(arr, size);

    cout << "The index of the smallest element is: " << smallestIndex << std::endl;

    return 0;
}

int index(const int arr[], int size) {
    if (size == 1) {
        return 0;
    }
    else {
        int smallestIndex = index(arr, size - 1);
        if (arr[smallestIndex] < arr[size - 1]) {
            return smallestIndex;
        }
        else {
            return size - 1;
        }
    }
}
```

#3

```cpp
#include <iostream>
using namespace std;


struct TreeNode {
    int val;
    TreeNode* left;
    TreeNode* right;
    TreeNode(int v) : val(v), left(nullptr), right(nullptr) {}
};

void insertTreeNode(TreeNode*& root, int val) {
    if (!root) {
        root = new TreeNode(val);
    }
    else if (val < root->val) {
        insertTreeNode(root->left, val);
    }
    else {
        insertTreeNode(root->right, val);
    }
}

void order(TreeNode* root) {
    if (root) {
        order(root->left);
        cout << root->val << " ";
        order(root->right);
    }
}

//#4

int main() {
    TreeNode* root = nullptr;

    insertTreeNode(root, 10);
    insertTreeNode(root, 5);
    insertTreeNode(root, 13);
    insertTreeNode(root, 9);
    insertTreeNode(root, 15);

    cout << "Tree: ";
    order(root);
    cout << endl;

    return 0;
}
```