***Question No.2***

1. **STL (Vectors):** **Your task is to develop an inventory management system for an online store. Use vectors to implement a function that adds new products to the inventory and another function that removes products based on their IDs.**

#include <iostream>

#include <vector>

Using namespace std;

struct Product {

int id;

string name;

Product(int i, const string& n) : id(i), name(n) {}

};

class Inventory {

private:

vector<Product> products;

public:

void addProduct(int id, const string& name) {

for (const Product& product : products) {

if (product.id == id) {

cout << "Product with ID " << id << " already exists." << std::endl;

return;

}

}

Product newProduct(id, name);

products.push\_back(newProduct);

cout << "Product added: ID - " << id << ", Name - " << name << endl;

}

void removeProduct(int id) {

auto it = find\_if(products.begin(), products.end(),

[id](const Product& p) { return p.id == id; });

if (it != products.end()) {

products.erase(it);

cout << "Product with ID " << id << " removed from inventory." << endl;

} else {

cout << "Product with ID " << id << " not found in inventory." << endl;

}

}

};

int main() {

Inventory inventory;

inventory.addProduct(101, "Product A");

inventory.addProduct(102, "Product B");

inventory.addProduct(103, "Product C");

inventory.removeProduct(102);

inventory.removeProduct(104);

return 0;

}

1. **STL (Algorithms):** **You are tasked with evaluating the performance of Bubble Sort against the widely-used STL sort algorithm. The goal is to analyze the execution time of each algorithm when sorting a large vector of 100,000 integers in ascending order (initialize the vector in descending order).**

#include <iostream>

#include <vector>

#include <algorithm>

#include <chrono>

Using namespace std;

void bubbleSort(vector<int>& arr) {

int n = arr.size();

for (int i = 0; i < n - 1; ++i) {

for (int j = 0; j < n - i - 1; ++j) {

if (arr[j] > arr[j + 1]) {

swap(arr[j], arr[j + 1]);

}

}

}

}

int main() {

const int size = 100000;

vector<int> data(size);

for (int i = 0; i < size; ++i) {

data[i] = size - i;

}

auto startBubble = std::chrono::steady\_clock::now();

bubbleSort(data);

auto endBubble = chrono::steady\_clock::now();

chrono::duration<double> timeBubble = endBubble - startBubble;

for (int i = 0; i < size; ++i) {

data[i] = size - i;

}

auto startStdSort = chrono::steady\_clock::now();

sort(data.begin(), data.end());

auto endStdSort = chrono::steady\_clock::now();

chrono::duration<double> timeStdSort = endStdSort - startStdSort;

cout << "Bubble Sort Time: " << timeBubble.count() << " seconds" << endl;

cout << "sort Time: " << timeStdSort.count() << " seconds" << endl;

return 0;

}