**QUESTION #2**

**INVENTORY MANAGEMENT:**

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

class Product

{

public:

int product\_id;

string product\_name;

double product\_price;

Product(int id, const string& name, double price)

: product\_id(id), product\_name(name), product\_price(price)

{}

int get\_product\_id() const

{

return product\_id;

}

const string& get\_product\_name() const

{

return product\_name;

}

double get\_product\_price() const

{

return product\_price;

}

};

class Inventory

{

private:

vector<Product> products;

public:

void add\_Product(int product\_id, const string& product\_name, double product\_price)

{

products.emplace\_back(product\_id, product\_name, product\_price);

}

void removeProduct(int product\_id)

{

products.erase(remove\_if(products.begin(), products.end(), [product\_id](const Product& p)

{

return p.product\_id == product\_id;

}), products.end());

}

void print\_Inventory() const

{

cout << "Inventory that is remaining: " << endl;

for (const auto& product : products)

{

cout << "ID of product: " << product.get\_product\_id() << ", Name of product: " << product.get\_product\_name() << ", Price of product: " << product.get\_product\_price() << endl;

}

}

};

int main()

{

Inventory inventory1;

inventory1.add\_Product(1, "Mobilephone", 545.3);

inventory1.add\_Product(2, "Laptop", 878.6);

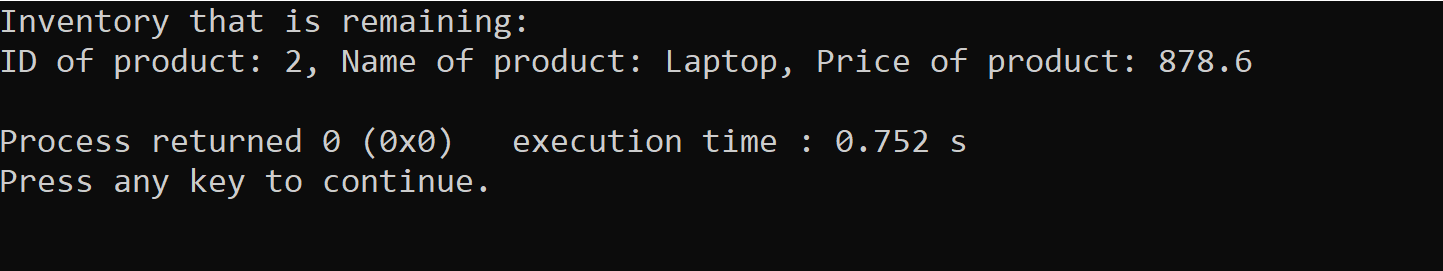
inventory1.removeProduct(1);

inventory1.print\_Inventory();

return 0;

}

**OUTPUT:**



**SORTING PERFORMANCE:**

**STL SORTING:**

#include <iostream>

#include <vector>

#include <algorithm>

#include <chrono>

using namespace std;

int main()

{

vector<int> arr(100000);

for (int i = 0; i < 100000; i++)

{

arr[i] = 100000 - i;

}

auto start = chrono::high\_resolution\_clock::now();

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

auto end =chrono::high\_resolution\_clock::now();

cout << "STL Sort execution time: " << chrono::duration\_cast<chrono::microseconds>(end - start).count() << " microseconds" << std::endl;

cout << "First 10 elements: ";

for (int i = 0; i < 10; i++)

{

cout << arr[i] << " ";

}

cout << endl;

cout << "Last 10 elements: ";

for (int i = 0; i < 10; i++)

{

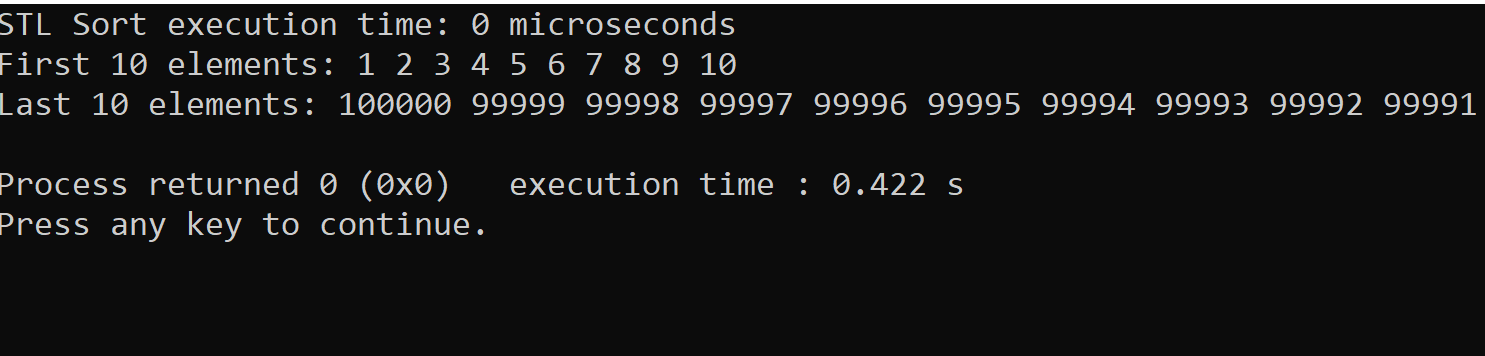
cout << arr[arr.size() - 1 - i] << " ";

}

cout << endl;

}

**OUTPUT:**



**BUBBLE SORTING:**

#include <iostream>

#include <vector>

#include <chrono>

using namespace std;

using namespace chrono;

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 = size; i > 0; --i) {

data.push\_back(i);

}

auto start = high\_resolution\_clock::now();

bubbleSort(data);

auto stop = high\_resolution\_clock::now();

auto duration = duration\_cast<milliseconds>(stop - start);

cout << "Bubble Sort Execution Time: " << duration.count() << " milliseconds\n";

cout << "First 10 integers: ";

for (int i = 0; i < 10; ++i) {

cout << data[i] << " ";

}

cout << "\n";

cout << "Last 10 integers: ";

for (int i = size - 10; i < size; ++i) {

cout << data[i] << " ";

}

cout << "\n";

}

**OUTPUT:**

