**Department of Engineering Technology**



**Foundation University Islamabad** **School of Science and Technology**

**DSA Lab**

**Name: Muhammad Usman**

**Roll no: 079**

**Lab#: 04**

**Topic: \_\_\_\_\_\_\_ OBJECTIVES:**

i. Objective - 1 ii. Objective - 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Performance** | |  | **Lab Report** | |  |
| **Description** | **Total**  **Marks** | **Marks Obtained** | **Description** | **Total**  **Marks** | **Marks** **Obtained** |
| Implementation of Code | 5 |  | Lab Exercises | 5 |  |
| **Total Marks obtained** | |  |  | |  |

**Q1:**

#include <iostream>

#include <string>

using namespace std;

// Define a structure for antique items

struct AntiqueItem {

string name;

string type;

string serialNumber;

};

const int MAX\_ITEMS = 100;

int sequentialSearch(AntiqueItem items[], int size, string targetSerial) {

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

if (items[i].serialNumber == targetSerial) {

return i; // found at index i

}

}

return -1; // not found

}

int main() {

int totalItems;

cout << "Enter total number of antique items in inventory: ";

cin >> totalItems;

AntiqueItem inventory[MAX\_ITEMS];

cin.ignore(); // Clear input buffer

// Input data for all antique items

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

cout << "\nEnter details for item " << i + 1 << ":\n";

cout << "Name: ";

getline(cin, inventory[i].name);

cout << "Type: ";

getline(cin, inventory[i].type);

cout << "Serial Number: ";

getline(cin, inventory[i].serialNumber);

}

// Serial number to search

string targetSerial;

cout << "\nEnter serial number of the vintage pocket watch to search: ";

getline(cin, targetSerial);

// Perform sequential search

int index = sequentialSearch(inventory, totalItems, targetSerial);

if (index != -1) {

cout << "\n? Pocket Watch Found!\n";

cout << "Name: " << inventory[index].name << endl;

cout << "Type: " << inventory[index].type << endl;

cout << "Serial Number: " << inventory[index].serialNumber << endl;

} else {

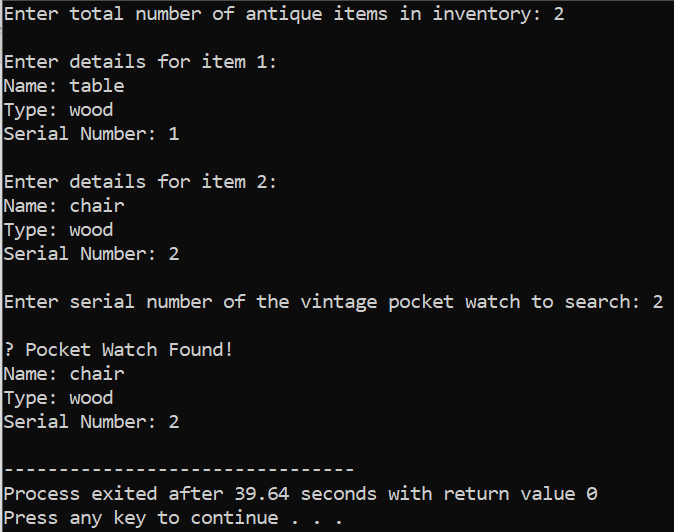
cout << "\n? Pocket Watch with Serial Number '" << targetSerial << "' not found in inventory.\n";

}

return 0;

}

**Output:**



**Q2:**

#include <iostream>

using namespace std;

// Define a structure for student records

struct Student {

int id;

string name;

string section;

};

// Binary search function to find student by ID

int binarySearch(Student students[], int size, int targetID) {

int low = 0, high = size - 1;

while (low <= high) {

int mid = (low + high) / 2;

if (students[mid].id == targetID)

return mid; // Found

else if (students[mid].id < targetID)

low = mid + 1;

else

high = mid - 1;

}

return -1; // Not found

}

int main() {

const int SIZE = 10;

// Sorted list of students by ID

Student students[SIZE] = {

{101, "Ali", "BSET-A"},

{102, "Ayesha", "BSET-B"},

{103, "Bilal", "BSET-C"},

{104, "Dua", "BSET-A"},

{105, "Fahad", "BSET-B"},

{106, "Hira", "BSET-C"},

{107, "Imran", "BSET-A"},

{108, "Komal", "BSET-B"},

{109, "Noman", "BSET-C"},

{110, "Usman", "BSET-B"}

};

int searchID;

cout << "Enter Student ID to search: ";

cin >> searchID;

int index = binarySearch(students, SIZE, searchID);

if (index != -1) {

cout << "\n✅ Student Found:\n";

cout << "ID: " << students[index].id << endl;

cout << "Name: " << students[index].name << endl;

cout << "Section: " << students[index].section << endl;

} else {

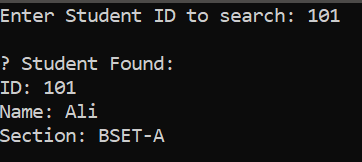
cout << "\n❌ Student with ID " << searchID << " not found.\n";

}

return 0;

}

**Output:**



**Q3:**

#include <iostream>

using namespace std;

int main() {

const int SIZE = 15;

int scores[SIZE];

// Input student scores

cout << "Enter scores of 15 students:\n";

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

cout << "Score of student " << i + 1 << ": ";

cin >> scores[i];

}

// Bubble Sort algorithm

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

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

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

// Swap

int temp = scores[j];

scores[j] = scores[j + 1];

scores[j + 1] = temp;

}

}

}

// Display sorted scores

cout << "\nStudent scores in ascending order:\n";

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

cout << "Rank " << i + 1 << ": " << scores[i] << endl;

}

return 0

;}

**Output:**

