|  |  |  |
| --- | --- | --- |
| **American University of Sharjah**  **College of Engineering**  Dept of Computer Science & Engg  P. O. Box 26666  Sharjah, UAE | A picture containing logo  Description automatically generated | **Instructors:** Dr.Aliaa Moualla  **Lab Instructor:** Sameer Alawnah  **Office:** EB1-0012C  **Phone**: 971-6-515-4940  **e-mail**: salawnah@aus.edu  **Semester**: Spring 2024 |

**CMP 220L - Programming II**

**Lab #2 –** **Vectors, Dynamic memory, and Structs**

**Note: using ChatGPT will be considered a violation of the AUS integrity code.**

**Objectives:**

* To practice using vectors.
* To practice passing vectors to functions.
* To practice 2D arrays
* To practice passing structs to functions.

Using Visual Studio 2022, write the below programs, compile and provide screenshots of output.

Note: you are required to submit copy of the code + screenshots of program run for each exercise.

**Exercise #1**

**Code 1:**

In healthcare system the administrative team needs a straightforward way to retrieve the unique patient ID for accessing patient records and carrying out essential administrative functions. Define a **linearSearch** function that performs a linear search on an array to find a target ID.

**int** **linearSearch(int arr[], int target)**

The function iterates through each element in the array and compares it to the target ID. If a match is found, it returns the index of the ID; otherwise, it returns -1 to indicate that the ID was not found.

In the main function, create an array of integers for the patient’s IDs, search for **patient with id 3456** using the **linearSearch**. Then print whether the desired patient ID was successfully found and display its corresponding index.

**int patientIDs[] = {1234, 2345, 3456, 4567, 5678, 6789, 7890, 8900};**

**Sample output**

Enter the target patient ID: 3456

Patient ID 3456 found at index 2

#include <iostream>

#include <vector>

**using** **namespace** std;

**int** linearSearch(**int** arr[], **int** size, **int** target) {

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

**if** (arr[i] == target) {

**return** i;

}

}

**return** -1;

}

**int** main() {

**int** patientIDs[] = {1234, 2345, 3456, 4567, 5678, 6789, 7890, 8900};

**int** size = **sizeof**(patientIDs) / **sizeof**(patientIDs[0]);

cout << "Enter the target patient ID: ";

**int** targetID;

cin >> targetID;

**int** result = linearSearch(patientIDs, size, targetID);

**if** (result != -1) {

cout << "Patient ID " << targetID << " found at index " << result << endl;

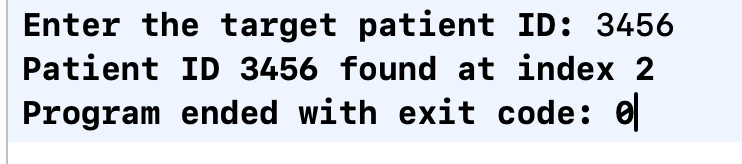
} **else** {

cout << "Patient ID " << targetID << " not found." << endl;

}

**return** 0;

}

****

**Code 2:**

Modify the above program so that the **linearSearch** function performs a linear search on a vector of integers to find a target ID.

**int** **linearSearch(const vector<int>& arr, int target)**

The function iterates through each element in the vector and compares it to the target ID. If a match is found, it returns the index of the ID; otherwise, it returns -1 to indicate that the ID was not found.

In the main function, create a vector of integers for the patient’s IDs search for patient with id 3456 using the **linearSearch**. Then print whether the desired patient ID was successfully found and display its corresponding index.

**vector<int> patientIDs = {1234, 2345, 3456, 4567, 5678, 6789, 7890, 8900};**

**Sample output**

Enter the target patient ID: 3456

Patient ID 3456 found at index 2

#include <iostream>

#include <vector>

**using** **namespace** std;

**int** linearSearch(**const** vector<**int**>& arr, **int** target) {

**for** (**int** i = 0; i < arr.size(); i++)

{

**if** (arr[i] == target)

{

**return** i;

}

}

**return** -1;

}

**int** main() {

**int** targetID;

vector<**int**> patientIDs = {1234, 2345, 3456, 4567, 5678, 6789, 7890, 8900};

cout << "Enter the target patient ID: ";

cin >> targetID;

**int** result = linearSearch(patientIDs, targetID);

**if** (result != -1) {

cout << "Patient ID " << targetID << " found at index " << result << endl;

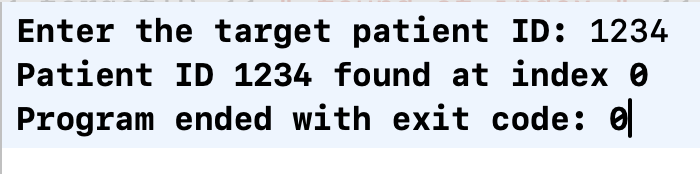
} **else** {

cout << "Patient ID " << targetID << " not found." << endl;

}

**return** 0;

}

****

**Exercise #2**

A 2D array in C++ is essentially an array of arrays. It's a matrix-like structure that allows you to store and manipulate data in rows and columns.

1. declare a 2D array of integers called **matrix** with 3 rows and 3 columns.
2. initialize the elements of the 2D array from 1 to 9 using nested for loops to traverse each row and column. The values are assigned based on the row and column indices.
3. We access and print the elements of the 2D array using nested for loops. The outer loop iterates over rows, and the inner loop iterates over columns.

**Sample output**

The elements of the array:

1 2 3

4 5 6

7 8 9

#include <iostream>

#include <vector>

**using** **namespace** std;

**int** main(){

**int** matrix[3][3];

**int** count = 1;

cout<<"The elements of the array...\n";

**for**(**int** i=0; i < 3; i++)

{

**for**(**int** j=0; j < 3; j++)

{

matrix[i][j] = count++;

cout << matrix[i][j] <<" ";

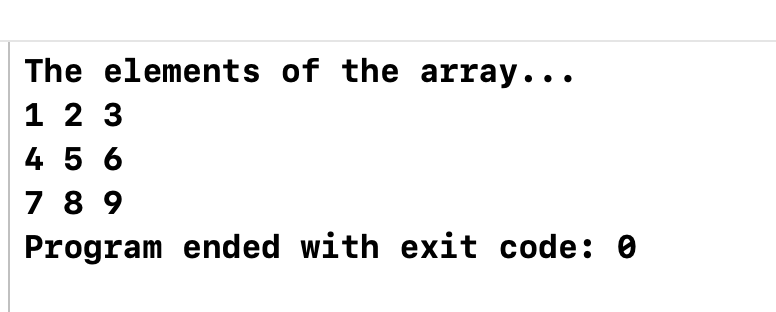
}

cout << endl;

}

**return** 0;

}

****

**Exercise #3**

Create a structure called **Movie** with the following attributes:

* name of type c-string (char[100])
* release\_year of type integer
* revenue of type float

Develop the following two functions:

* **void readMovie(Movie &m);** this function should ask the user and read the following:
  + The name of the movie m, as one full line.
  + The release\_year of the movie m.
  + The revenue of the movie m.
* **void printMovie(const Movie &m);** this function should print the name, release\_year and the revenue of the movie.

In the main, create an object (mov) of type movie, then call the readMovie and printMovie to read a movie and print it back.

**Sample output**

Enter movie name: Avatar: The Way of Water

Enter Release Year: 2022

Enter Revenue: 684075767

Name {Avatar: The Way of Water} Release Year {2022} Revenue {6.84076e+08}

#include <iostream>

#include <vector>

**using** **namespace** std;

**struct** Movie{

**char** name[100];

**int** release\_year;

**float** revenue;

};

**void** readMovie(Movie &m)

{

cout<<"Enter movie name: ";

cin.getline(m.name, 100);

cout<<"Enter Release Year: ";

cin >> m.release\_year;

cout<<"Enter revenue: ";

cin >> m.revenue;

cin.ignore();

}

**void** printMovie(**const** Movie &m)

{

cout << "Name {" <<m.name <<"} Release Year {" << m.release\_year <<"} Revenue {" << m.revenue << "}";

}

**int** main(){

Movie mov;

readMovie(mov);

printMovie(mov);

**return** 0;

}

**A close-up of a text

Description automatically generated**