/\*

Un establecimiento educativo que se encuentra lanzando una serie de cursos,

requiere de una aplicación que le permita inscribir a los participantes.

Para ello cuenta con una cantidad M de cursos a brindar a la comunidad.

Desalumnoollar una aplicación que permita:

\* Registrar una cantidad M de cursos. El código de curso, deberá asignarse de manera automática

y debe ser un número entero en el rango [1, M].

\* Registrar una cantidad de N alumnos/as. Un alumno/a sólo puede inscribirse en un curso.

\* Imprimir los datos de un alumno/a y el curso al que se inscribió.

\* Para obtener los datos, se ingresará el dni de la persona.

Restricciones:

\* El conjunto de cursos no puede ser recorrido secuencialmente.

\* El conjunto de alumnos/as sólo puede ser recorrido secuencialmente 1 vez.

\*/

#include <iostream>

using namespace std;

typedef struct {

string nombre;

int cantidadDeHoras;

} ST\_CURSO;

typedef struct {

int dni;

string nombre;

int edad;

string email;

string telefono;

int curso; // 1,2,...M

} ST\_ALUMNO;

int busquedaBinaria(ST\_ALUMNO alumno[], int n, int valorBuscado){

// Cuando alumnoanco evalúo todo el vector de 0 a N-1

int inicio = 0;

int final = n-1;

while (final >= inicio){ // la condicion indica que el vector no se haya recorrido en su totalidad

int mitad = inicio + (final - inicio) / 2;

// Si el elemento es el del medio, devolvemos la posición

if (alumno[mitad].dni == valorBuscado){

return mitad;

}

// Si el elemento es menor entonces solo puede estar en la primer mitad

if(alumno[mitad].dni > valorBuscado){

final = mitad - 1; // acota el limite superior

} else {

inicio = mitad + 1; // acota limite inferior

}

}

// Si salimos por acá, es que el valor no estaba

return -1;

}

void burbujeo(ST\_ALUMNO arr[], int n){

int i, j, aux;

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

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

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

aux = arr[j].dni;

arr[j].dni = arr[j+1].dni;

arr[j+1].dni = aux;

}

}

}

}

void cargarCurso(ST\_CURSO array[], int cant) {

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

cout << "Ingrese el codigo del curso: ";

cin >> codigoCurso;

cout << "Ingrese el nombre del curso: ";

cin >> array[codigoCurso -1 ].nombre;

cout << endl;

cout << "Ingrese la cantidad de horas: ";

cin >> array[codigoCurso -1].cantidadDeHoras;

cout << endl;

}

return;

}

void cargarAlumnos (ST\_ALUMNO array[], int cant) {

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

cout << "Ingrese DNI" << endl;

cin >> alumno[i].dni;

cout << "Ingrese nombre" << endl;

cin >> alumno[i].nombre;

cout << "Ingrese edad" << endl;

cin >> alumno[i].edad;

cout << "Ingrese email" << endl;

cin >> alumno[i].email;

cout << "Ingrese teléfono" << endl;

cin >> alumno[i].telefono;

cout << "Ingrese curso" << endl;

cin >> alumno[i].curso;

return;

}

/\*

codigo i+1

\*/

int main () {

int n, m;

cout << "Ingrese la cantidad de cursos a lanzar: " << endl;

cin >> m;

ST\_CURSO curso[m];

cout << "Ingresar cantidad de alumnos" << endl;

cin >> n;

ST\_ALUMNO alumno[n];

// Funcion para cargar cursos

cargarVector(curso, m);

// Ciclo para cargar alumnos

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

cout << "Ingrese DNI" << endl;

cin >> alumno[i].dni;

cout << "Ingrese nombre" << endl;

cin >> alumno[i].nombre;

cout << "Ingrese edad" << endl;

cin >> alumno[i].edad;

cout << "Ingrese email" << endl;

cin >> alumno[i].email;

cout << "Ingrese teléfono" << endl;

cin >> alumno[i].telefono;

cout << "Ingrese curso" << endl;

cin >> alumno[i].curso;

/\*

while (alumno[i].curso < 1 && alumno[i].curso > m){

cout << "Curso inválido, ingrese un curso del 1 al " << m << endl;

cin >> alumno[i].curso;

}

\*/

}

// Ordenamiento

burbujeo(alumno, n);

// Busqueda alumno por DNI

int valorBuscado;

cout << "Ingrese DNI para buscar alumno" << endl;

cin >> valorBuscado;

int posicion = busquedaBinaria(alumno, n, valorBuscado);

// Impresión

if(posicion >= 0){

int posicionCurso = alumno[posicion].curso;

cout << "DNI del alumno: " << endl;

cout << alumno[posicion].dni << endl;

cout << "Nombre del alumno: " << endl;

cout << alumno[posicion].nombre << endl;

cout << "Edad del alumno: " << endl;

cout << alumno[posicion].edad << endl;

cout << "Email del alumno: " << endl;

cout << alumno[posicion].email << endl;

cout << "Telefono del alumno: " << endl;

cout << alumno[posicion].telefono << endl;

cout << "Numero de curso del alumno: " << endl;

cout << alumno[posicion].curso << endl;

cout << "Nombre de curso del alumno: " << endl;

cout << curso[posicionCurso - 1].nombre << endl;

} else {

cout << "El DNI ingresado no corresponde a un alumno registrado" << endl;

}

return 0;

}

estoy pasando a funcion la carga de alumnos

y ya modifique el procedimiento de cargar curso ahora lo paso

yo también lo estoy arreglando. Muy bien parece que nos fue. Creo que fue clave separarnos entre el discord y zoom para laburar tranquilos

Grande gonza que tenia todo copiado de la clase. Yo no tenía ni una línea de codigo de lo de hoy

si, felicitaciones chicos! hicieron genial lo de búsqueda y burbujeo. Luego voy a mirar bien el código para terminar de organizar ideas.

Coincido lo de dividir Discord y Zoom.. Grande Gonza!

ok, ahora entonces lo copio cuando esté terminado.

Fino. Sino no alcanzaba el tiempo, nos pasaría como en el parcial

Me causó gracia lo de Dream Team, les paso un link de Discord por si quieren que sigamos hablando por ahí o para otros ejercícios para resolver en equipo.

Gonza Álvarez, venís? <https://discord.gg/cVdbnYwQ> =)P

Chicos, ya está el código o van a modificar algo más?