using System;

using System.Collections.Generic;

using System.Text;

using System.Text.RegularExpressions;

using static System.Runtime.InteropServices.JavaScript.JSType;

namespace GroupAssignment

{

// Reema Alghamdi

class Attendee

{

private string name;

private int age;

private string specialization;

private int id;

public Attendee(string n, int a, string special, int id)

{

Name = n;

Age = a;

Specialization = special;

Id = id;

}

public string Name

{

get { return name; }

set

{

if (value != null)

name = value;

else

throw new Exception("Name must not be null");

}

}

public int Age

{

get{ return age; }

set

{

if(value > 0)

age = value;

else

throw new Exception("Age must be > 0");

}

}

public string Specialization

{

get { return specialization; }

set

{

if(value.Equals("web design") || value.Equals("programming") || value.Equals("acritical inelegant"))

specialization = value;

else

throw new Exception("The specialization isn't valid ");

}

}

public int Id

{

get { return id; }

set

{

if (value > 0) id = value;

else

throw new Exception(" ID must be >0 ");

}

}

public override string ToString()

{

return string.Format(" Name: {0}, Age: {1}, ID: {2}, specialization: {3} ", Name, Age, Id, specialization);

}

} // end class Attendee

// Leena Almatar

abstract class ScientificEvent

{

private string title;

private string location;

private int maxTicket;

private int id;

Attendee[] atds;

int count = 0;

public ScientificEvent(string t, string l, int mt, int i)

{

Title = t;

Location = l;

MaxTicket = mt;

Atds = new Attendee[MaxTicket];

Id = i;

}

public string Title

{

get { return title; }

set

{

if(value != null)

title = value;

else

throw new Exception("Value must not be null");

}

}

public string Location

{

get { return location; }

set

{

if(value != null)

location = value;

else

throw new Exception("Value must not be null");

}

}

public int MaxTicket

{

get { return maxTicket; }

set

{

if(value > 0)

maxTicket = value;

else

throw new Exception("Value must be > 0");

}

}

public int Id

{

get { return id; }

set

{

if(value > 0)

id = value;

else

throw new Exception("Value must be > 0");

}

}

public Attendee[] Atds

{

get { return atds; }

set { atds = value; }

}

public abstract bool acceptAttendee(Attendee a);

public void addAttendee(Attendee a)

{

bool found = false;

if (count < Atds.Length)

{

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

{

if (a.Id == Atds[i].Id)

{

found = true;

break;

}

}

if (!found && acceptAttendee(a))

{

Atds[count] = a;

count++;

Console.WriteLine("Reservation successful");

}

else

Console.WriteLine("Sorry, there are a problem with the reservation process");

}

else

Console.WriteLine("Sorry,theeventfull..");

}

public bool removeAttendee(Attendee a)

{

bool found = false;

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

{

if (Atds[i] != null)

{

if (a.Id == Atds[i].Id)

{

Atds[i] = null;

count--;

found = true;

break;

}

}

}

return found;

}

public override string ToString()

{

string a = "";

for (int i = 0; i < Atds.Length; i++)

{

if(Atds[i] != null)

a += Atds[i];

}

return string.Format("Title: {0}, Location: {1}, ID: {2}, Number of Attendances is: {3}, List of Attendees:{4}", Title, Location, Id, maxTicket, a);

}

} // end class ScintificEvent

// Ruba Alhudyani and file coordinate

class Workshop : ScientificEvent

{

private string track;

private int nbWeeks;

private string[] speakers;

public Workshop(string t, string l, int mt, int i, string track, int nbWeeks, string[] speakers)

: base(t, l, mt, i)

{

this.Track = track;

this.NbWeeks = nbWeeks;

this.speakers = speakers;

}

public string Track

{

get { return track; }

set

{

if (value.Equals("web design") || value.Equals("programming") || value.Equals("acritical inelegant"))

track = value;

else

throw new Exception("Track is not valid ");

}

}

public int NbWeeks

{

get { return nbWeeks; }

set

{

if (value > 0)

nbWeeks = value;

else

throw new Exception("The value must be > 0");

}

}

public string[] Speakers

{

get { return speakers; }

set

{

speakers = value;

}

}

public override bool acceptAttendee(Attendee a)

{

if (a.Specialization.Equals("web design") || a.Specialization.Equals("programming") || a.Specialization.Equals("acritical inelegant"))

return true;

else

return false;

}

public override string ToString()

{

string s = "";

for (int i = 0; i < Speakers.Length; i++)

{

if (speakers[i] != null)

s += speakers[i];

}

return base.ToString() + " Track: " + Track + " Number of weeks:" + NbWeeks + " Speakers:" + s;

}

} // end class workshop

// Nawir Alsahli

class Training : ScientificEvent

{

private string profName;

public Training(string t, string l, int mt, int i, string pname) : base(t, l, mt, i)

{

ProfName = pname;

}

public string ProfName

{

get { return profName; }

set { profName = value; }

}

public override bool acceptAttendee(Attendee a)

{

if (a.Age > 12) return true;

else

return false;

}

public override string ToString()

{

return base.ToString() + " and Professor Name: " + ProfName;

}

} // end class training

// Yara Alghamdi

class Program

{

static void Main(string[] args)

{

ScientificEvent[] events;

Console.WriteLine("How many events you want to insert?");

int size = Convert.ToInt32(Console.ReadLine());

events = new ScientificEvent[size];

int eventCount = 0;

int selection = 0;

while (selection != 5)

{

Console.WriteLine("1. Add new workshop, and add all attendances" +

"\n2. Add a new training to the array, and add all attendances" +

"\n3. Print the sum of maximum number of attendances (maxTicket) in all the workshops" +

"\n4. Print the information of all training events that have the same professor’s name." +

"\n5. Exit");

selection = Convert.ToInt32(Console.ReadLine());

switch (selection)

{

case 1:

Console.WriteLine("Enter the title of the event:");

string title = Console.ReadLine();

Console.WriteLine("Enter the location of the event:");

string location = Console.ReadLine();

Console.WriteLine("Enter the max number of ticket for the event:");

int max = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter ID of the event:");

int id = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter Track in this workshop: ");

string track = Console.ReadLine();

Console.WriteLine("Enter Number of weeks in this workshop:");

int NumberOfWeeks = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter the number of speakers in this workshop: ");

int NumberOfSpeakers = Convert.ToInt32(Console.ReadLine());

string[] speakers = new string[NumberOfSpeakers];

Console.WriteLine("Enter names of speakers in this workshop: ");

for (int i = 0; i < speakers.Length; i++)

{

speakers[i] = Console.ReadLine();

}

ScientificEvent workshopEvent = new Workshop(title, location, max, id, track, NumberOfWeeks, speakers);

Console.WriteLine("How many attendee in this workshop:");

int attendeCount = Convert.ToInt32(Console.ReadLine());

if(attendeCount <= max)

{

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

{

Console.WriteLine("Enter attendee name:");

string name = Console.ReadLine();

Console.WriteLine("Enter attendee age:");

int age = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter attendee specialization:");

string specialization = Console.ReadLine();

Console.WriteLine("Enter attendee id:");

int attendeeId = Convert.ToInt32(Console.ReadLine());

workshopEvent.addAttendee(new Attendee(name, age, specialization, attendeeId));

}

}

else

Console.WriteLine("Number of attendee must be less than " + max);

if(eventCount < size)

{

events[eventCount] = workshopEvent; eventCount++;

}

else

Console.WriteLine("event List is full.");

break;

case 2:

Console.WriteLine("Enter the title of the event:");

title = Console.ReadLine();

Console.WriteLine("Enter the location of the event:");

location = Console.ReadLine();

Console.WriteLine("Enter the max number of ticket for the event:");

max = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter ID of the event:");

id = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter Prof name of this training:");

string profname = Console.ReadLine();

Training trainingEvent = new Training(title, location, max, id, profname);

Console.WriteLine("How many attendee in this workshop:");

attendeCount = Convert.ToInt32(Console.ReadLine());

if (attendeCount < max)

{

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

{

Console.WriteLine("Enter attendee name:");

string name = Console.ReadLine();

Console.WriteLine("Enter attendee age:");

int age = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter attendee specialization:");

string specialization = Console.ReadLine();

Console.WriteLine("Enter attendee id:");

int attendeeId = Convert.ToInt32(Console.ReadLine());

trainingEvent.addAttendee(new Attendee(name, age, specialization, attendeeId));

}

}

else

Console.WriteLine("Number of attendee must be less than " + max);

if (eventCount < size)

{

events[eventCount] = trainingEvent;

eventCount++;

}

else

Console.WriteLine("event List is full!");

break;

case 3:

int sum = 0;

for (int i = 0; i < events.Length; i++)

{

if (events[i] is Workshop)

sum = sum + events[i].MaxTicket;

}

Console.WriteLine("The sum of maximum number of attendances is:" + sum);

break;

case 4:

Console.WriteLine("Enter professor name:");

profname = Console.ReadLine();

for (int i = 0; i < events.Length; i++)

{

if (events[i] is Training)

{

Training t = (Training)events[i];

if (t.ProfName.Equals(profname))

Console.WriteLine(t);

}

}

break;

case 5:

break;

default:

Console.WriteLine("Invalid Choice!!");

break;

}

}

Console.ReadKey();

}

}

}