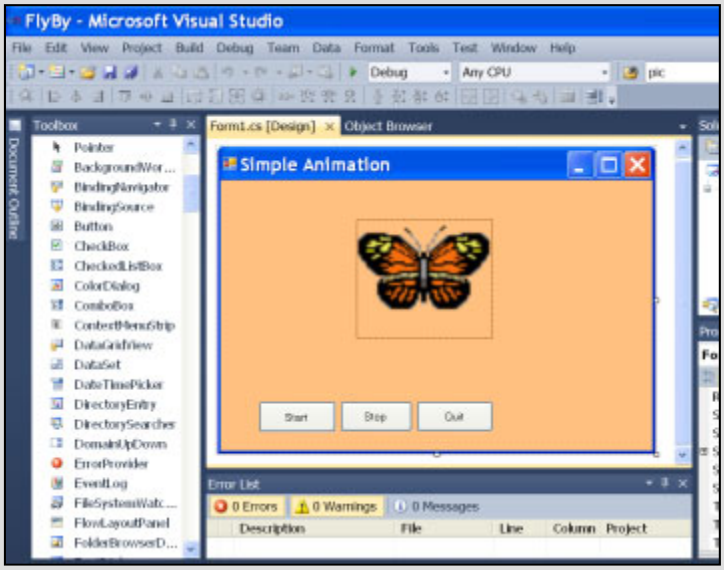
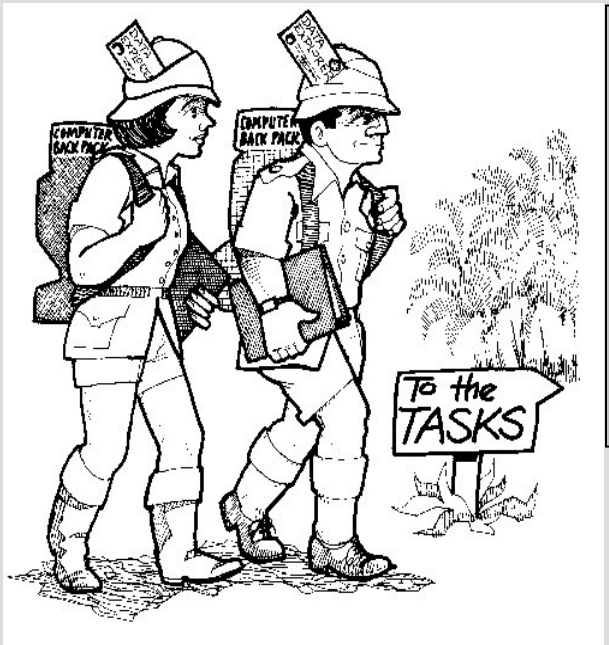
Windows Programming

bucks new university

Console & Windows Programming Using C#





Application Programming (CO453)

Part A Weeks 1-3

Unit 5: Arrays

Classwork (3 Tasks)

5.1 Tournament Scores

Look at project **Task6_1.csproj**. Compile and run it.

- Look at the code below
- Notice that the program has one class called **Tournament**.
- It defines an integer **array** called **scores** to hold all the scores in the tournament.
- The constructor for the class then actually creates the array of the required size.

```
class Tournament
{
    int[ ] scores;              // define scores as an integer array
    const int MAX = 6;          // set a constant size

    public static void Main()           // program starts executing here
    {
        Tournament  myTournament = new Tournament();  // create a new object
        myTournament.getScores();                     // call its getScores

    public Tournament()                 // the class constructor
    {
        scores = new int[MAX];          // create a new array of size MAX
    }

    public void getScores()
    {
        Console.WriteLine("Inputting the Tournament Scores");
        Console.WriteLine("========================");
        for (int i = 0; i < MAX; i++)
        {
            Console.Write("Enter score number " + (i + 1) + " : ");
            scores[i] = Convert.ToInt32(Console.ReadLine());
        }
    }

} // end of Tournament class
```

Task 5.1

1. Add another method to the class, called **showScores()** .. this should clear the screen and then display all the scores in the form:

   Tournament Scores
   ==========
   Player 1 scored < >
   Player 2 scored < > etc.

2. Change the size of the tournament to **12**. Check that it still works OK.
3. Put source code and sample outputs in your logbook

5.2 MP3 Chart Voter

Look at project **Task6_2.csproj**. Compile and run it.
The program presents you with a list of song tracks and you have to vote for your favourite.
Of course it is far from finished!

- Examine the existing code on the next page
- modify the program to complete these tasks:

Task 5.2

The program has to keep a count of all the votes for each song track.

1. Create a new integer array called **votes** that will be used to count the votes for each track
2. Now inside the **getVotes()** method you need to add one to the appropriate vote in the **votes** array. There are several ways of doing this. e.g. if the vote was for track 5 then add 1 to **votes[4]** (remember arrays start counting from 0)
3. Get the program to repeat for many voters by using a loop inside the **run()** method
4. Add another method called **showVotes()** which displays all the vote counts like this:

   MP3 Track Votes
   ==========
   Track 1 had < > votes
   Track 2 had < > votes etc.

   Total Number of Votes : < >
5. Expand the program to work for 10 tracks and add your own favourites to the list. Check that everything works OK.
6. Use **MAX** inside the **getVotes()** method so that it always asks you to vote correctly depending on the number of tracks e.g. Choose 1 – 10 etc.

Extra

- Can you get **showVotes()** to display the track titles as well as the votes?

Put source code and sample outputs in your logbook

```
class Mp3Chart
{
    string[ ]  topTen;          // define a string array called topTen
    const int MAX = 5;

    public static void Main()          // program starts executing here
    {
        Mp3Chart  myChart = new Mp3Chart();          // create new object
        myChart.run();                               // call its run method
    }

    public Mp3Chart()          // constructor
    {
        topTen = new string[MAX];        // create a new array of correct
size
        topTen[0] = "Revolution";  // initialise the array values
        topTen[1] = "Mera Dil Tuta Hain";
        topTen[2] = "CandyMan";
        topTen[3] = "Ruby Tuesday";

    public void run()
    {
        showMusicList();
        getVotes();
    }

    public void showMusicList()
    {
        Console.Clear();
        Console.WriteLine("\tMusic List");
        Console.WriteLine("\t==========");
        for (int i=0; i < MAX; i++)
        {
            Console.WriteLine("\tSong " + (i + 1) + " is " + topTen[i]);
        }
    }

    public void getVotes()
    {
        int  userVote;
        Console.WriteLine("\tSelect your favourite Song");
        Console.WriteLine("\t=========================");
        Console.Write("\tChoose 1 - 5 : ");
        userVote = Convert.ToInt32(Console.ReadLine());
    }

} // end of Mp3Chart class
```

5.3 Tournament Names

Modify the Tournament class of 5.1 as follows to deal with names as well as scores

1. Add a string array called **names** to the class.
2. Change the name of the **getScores()** method to **getDetails()** and use it to input all the names as well as the scores, like this:

   Input Names and Scores
   ==========
   Enter player 1 name : < >
   Enter player < > score : < >
   Enter player 2 name : < >
   Enter player < > score : < > etc.

3. Add a new method called **showDetails()** and use it to display all the names and scores, like this:

   Tournament Results
   ==========
   Player < > scored < >
   Player < > scored < > etc.

4. Add a new method called **showBest()** .. it should look through the scores to find the highest score and then print out the **name** and **score** for this person.

Put source code and sample outputs in your logbook

Independent Study (3 Tasks)

The following exercises are to be done individually and independently, in your own time.

5.4 Sorting

Computers spend a lot of time sorting things into order and there are many different sorting algorithms to choose from. One of the simplest (and slowest) is called the **Bubble Sort**. It has one loop contained inside another loop as shown here.

Bubble Sort for N items

```
loop N times
    loop from 0 up to N-1
        if current item > next item
            swap the two items
        end if
    end loop
end loop
```

Your Tasks

1. Create a new project for this task with a class called **Bubble**
2. Add a new method called **inputNumbers()** which inputs 6 numbers into an array
3. Add a second method called **display()** which clears the screen and displays all the numbers one above the other.
4. Make sure the program works correctly so far
5. Now add a third method called **sortNumbers()** which applies the Bubble Sort algorithm to sort the numbers into numerical order
6. Call the methods in the right order and get the sorting to work.
7. Try it for 20 numbers
8. Add 3 more methods to apply a similar technique to sorting a list of names into alphabetical order.

5.5 Traffic Survey

It has been decided to do a **traffic survey** at a particularly busy section of road.
Traffic is counted automatically during **24** 1-hour time periods in a typical day and the counts are then stored in an array in the program for later analysis.
You are to simulate this using an **array** for the 24 periods.

- Create a new project with a class called **Traffic**.
- Set up an integer array called **trafficCount** with 24 elements.
- Define a method called **enterCounts()** which allows the user to enter 24 counts into the array.
- Another method called **showTotal()** should calculate and display the total number of cars in the array.
- A third method called **busiest()** should work out and display the busiest time of day.
- A forth method called **showData()** should output **all** the data in a suitable table with the percentage of the total.
- Provide a **report()** method that does the following:
    - calls **enterCounts()**
    - calls **showData()**
    - calls **showTotal()**
    - calls **busiest()**

Your results should look something like the following:

```
Traffic Report
------------------
Hour        Car Count        Percentage of Total
-----------------------------------------------------------------
1           1200              7.7%
2           1155              6.4%
etc.

Total Car Count for the day = 15546
Busiest hour = 7

-----------------------------------------------------------------
```

Put source code and sample outputs in your logbook

5.6 The Bates Motel

Look at project **Task7_2.csproj**. Compile and run it. This simulates an incomplete booking system for the **Bates Motel**. Your task is improve the functionality of the program.

This is a **menu-based** program for booking and vacating rooms.

There are 5 options available on the repeating menu:

1. Book a room
2. Vacate a room
3. Display ALL Room Details
4. Vacate ALL rooms
5. Quit

But only item 1 (Book a Room) is currently implemented

NOTES

- The motel has 20 rooms.
- an **integer array** called **rooms** is used to store the number of guests in each room.
- Notice the size of this array has been set to **MAX+1** so we can use room numbers 1 to 20

Task 5.6

1. Start by implementing item **3** of the menu. Do this with a method called **showAllRooms()** This should display all the room details as follows:

   Bates Motel Room Status
   ==========
   Room 1 0 guests
   Room 2 2 guests etc.

2. Implement menu item **2** with a method called **vacateOneRoom()** which asks you which room you want to vacate and then puts 0 into this position of the array
3. Now implement menu item **4** which should allow you to vacate all the rooms. Use a method called **vacateAll()** which should put 0 into every position in the rooms array.
4. Check that all is working correctly.

Put source code and sample outputs in your logbook

Extra: The Bates Motel (contd)

1. Notice that you can currently double-book a room in the motel. If you choose a room that is already booked, the old booking is overwritten! Add some code to prevent this from happening in a user-friendly way.
2. Add a new Item **5** : **Management Information** on the menu. This should use a suitable method and provide useful information such as how many rooms are booked, how many guests are in the hotel and also the room numbers of all the empty rooms.
3. It is possible to book any number of guests into a room but the room limit is 4. Provide a user-friendly mechanism which only allows up to 4 guests per room.

```
class Motel
{
    int[ ]  rooms;                  // define an integer array called rooms
    const int MAX = 21:

    public static void Main()       // program starts executing here
    {
        Motel  BatesMotel = new Motel();        // create new object BatesMotel
        BatesMotel.runMotel();                  // call its runMotel

    public Motel()          // constructor
    {
        rooms = new int[MAX];       // allow room numbers from 1 to

    public void runMotel()
    {
        string choice = "";
        do
        {
            Console.Clear();
            Console.WriteLine("The Bates Motel");
            Console.WriteLine("===============");
            Console.WriteLine("1. Book a room");
            Console.WriteLine("2. Vacate a room");
            Console.WriteLine("3. Display ALL Room Details");
            Console.WriteLine("4. Vacate ALL rooms");
            Console.WriteLine("5. Quit");
            Console.Write("Enter your choice : ");
            choice = Console.ReadLine();
            if (choice == "1")
            {
                bookRoom();
            }

    public void bookRoom()
    {
        int  roomNumber, guests;
        Console.WriteLine("\nThe Bates Motel");
        Console.WriteLine("===============");
        Console.WriteLine("Book a room");
        Console.Write("Enter the room number : ");
        roomNumber = Convert.ToInt32(Console.ReadLine());
        Console.Write("How many guests : ");
        guests = Convert.ToInt32(Console.ReadLine());
        rooms[roomNumber] = guests;     // make the booking
        Console.WriteLine("Room " + roomNumber + " booked for " +

} // end of Motel class
```