

## Contents

<b>Solution</b>	<b>1</b>
Simplest Solution . . . . .	1
Using Classes . . . . .	3

## Solution

### Simplest Solution

A possible solution, using arrays but not resizing them, is as follows:

using System;

```
public class Program
{
    public static void Main(string[] args)
    {
        // Variable declarations.

        string[] todo = new string[100]; // This will hold
        ↪ the items in the todo list.
        // Note that we are arbitrarily deciding that the
        ↪ maximum number of items is 100.
        bool[] status = new bool[100]; // This will hold
        ↪ the status of each item.
        // true means "done", false means "not done".
        string uInput; // This will hold user input.
        int todoSize = 0; // This will hold the actual
        ↪ number of items in the list.
        int completed = 0; // This will hold the number of
        ↪ items done.
        int justdone; // This will hold the number of the
        ↪ last item completed.
        bool valid; // This will hold true if the user
        ↪ input is valid (a positive number
        // less than the number of items in the list),
        ↪ false otherwise. Used for user-input
        // validation.
        char itemStatus; // This will hold '☑' if the
        ↪ current item is done,
        // '☐' otherwise.

        // We start by populating the list with items.
        do
```

```

    {
        Console.WriteLine("What is on your todo list?
↪ Enter \"done\" when you are done.");
        uInput = Console.ReadLine();
        if (uInput != "done")
        {
            todo[todoSize] = uInput; // We can store
↪ the first item at index todoSize
            // since its initial value is 0.
            todoSize++; // We increment the number of
↪ items in the list.
        }
    } while (uInput != "done"); // When the user
↪ enters "done", we exit this loop.

    // We now display the todo list, and ask the user
↪ to indicate which item they
    // completed, as long as there are some items left
↪ in their list.

    while (completed != todoSize)
    {
        // We display the todo list.
        Console.WriteLine("Here is your current todo
↪ list:");
        Console.WriteLine("| # | Status | Task |");
        for (int i = 0; i < todoSize; i++)
        {
            if (status[i]) { itemStatus = '☑'; } else
↪ { itemStatus = '☐'; }
            Console.WriteLine("| "+ (i+1) + " | " +
↪ itemStatus + " | " + todo[i]);
        }
        // We now ask the user to enter the number of
↪ the completed item.
        valid = false; // We assume that the user has
↪ not given a valid value yet.
        do
        {
            Console.WriteLine("Enter the number of the
↪ task you completed.");
            valid = int.TryParse(Console.ReadLine(),
↪ out justdone) && 0 < justdone && justdone <= todoSize;
        } while (!valid);
    }

```

```

        status[justdone - 1] = true; // We indicate
↪ that the item was completed by setting its value to
↪ true.
        completed++; // We increment the number of
↪ items completed.
        Console.WriteLine($"You are {completed /
↪ (double)todoSize:P} done!");
        // Note that we force double division using
↪ casting, and use the :P format specifier.
    }
    Console.WriteLine("Congratulations!");
}
}
}

```

You can download it here<sup>1</sup>

## Using Classes

Another solution is to create a class for “todo list items” and to create an array of them. That is, have a class file `Todo.cs` along the lines of

```

class Todo{
    public string Description{get; set;}
    public bool Status{get; set;}
}

```

and then to create and manipulate arrays of `Todo` objects, for example as follows:

```

Todo[] todoList = new Todo[100];
todoList[0] = new Todo();
todoList[0].Description = "My first item";
todoList[0].Status = false;
Console.Write(todoList[0].Description +
↪ (todoList[0].Status ? " done" : " not done"));

```

---

<sup>1</sup>[https://princomp.github.io/code/projects/ToDoList\\_Array.zip](https://princomp.github.io/code/projects/ToDoList_Array.zip)