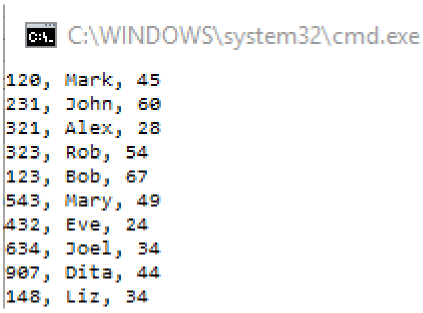
Lab 05: Working with Structure

In this exercise, we are to create a console application that creates a customised array called a structure. A structure differs from an array in terms of variable types it can hold. Unlike an array of variable (which must be of the same variable type), a structure can hold different variable types within the same declared variable name.

We will use the same sample output from the previous exercise (Declaring an Array). Although the output may be the same, the technique of achieving this will differ. The concept of accessing the values using its index number, however, remains the same.



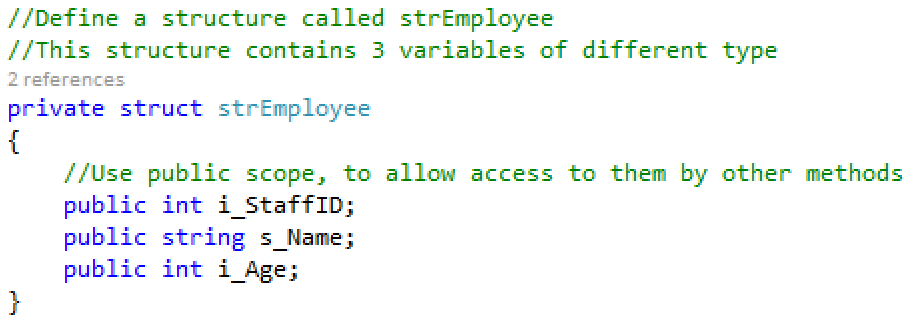
C:\WINDOWS\system32\cmd.exe

```
120, Mark, 45
231, John, 60
321, Alex, 28
323, Rob, 54
123, Bob, 67
543, Mary, 49
432, Eve, 24
634, Joel, 34
907, Dita, 44
148, Liz, 34
```

Defining a Structure

Structure in program is a customised variable that we create. It does not exist until we define them. Keyword to define a structure is *struct*.

Note that this only defines the format (or template) of the structure, not declaring it for use yet. Declaring for use needs to be done separately.

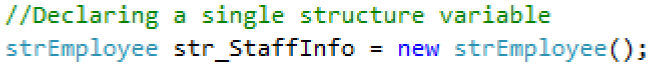


```
//Define a structure called strEmployee
//This structure contains 3 variables of different type
2 references
private struct strEmployee
{
    //Use public scope, to allow access to them by other methods
    public int i_StaffID;
    public string s_Name;
    public int i_Age;
}
```

Single Structure Variable

Declaring and initialising a single structure variable

A structure needs to be declared and initialised, similar to any other objects.

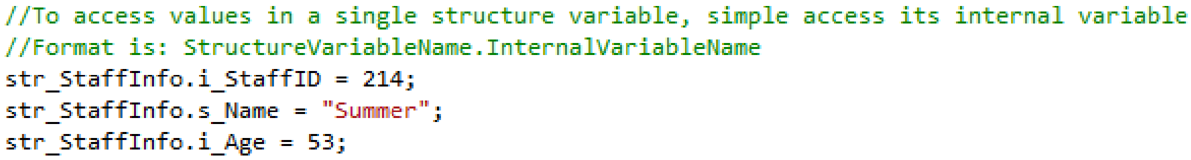


```
//Declaring a single structure variable
strEmployee str_StaffInfo = new strEmployee();
```

Accessing values in a single structure variable

Single structure variable can be access directly by specifying its internal variable names. This works the same way for storing and retrieving values.

The format is *StructureVariableName.InternalVariableName*.

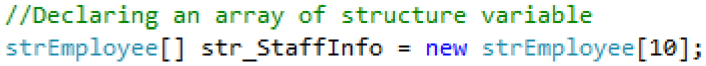


```
//To access values in a single structure variable, simple access its internal variable
//Format is: StructureVariableName.InternalVariableName
str_StaffInfo.i_StaffID = 214;
str_StaffInfo.s_Name = "Summer";
str_StaffInfo.i_Age = 53;
```

Array of Structure Variable

Declaring and initialising an array of structure variable

Structure contains multiple variables of different type within. This is useful when storing data of various types, grouping them under a common structure variable name. When dealing with multiple entries, we can create an array of a structure variable. The process is similar to declaring an array of built-in variable, i.e. adding a postfix *[ ]* to the structure variable.

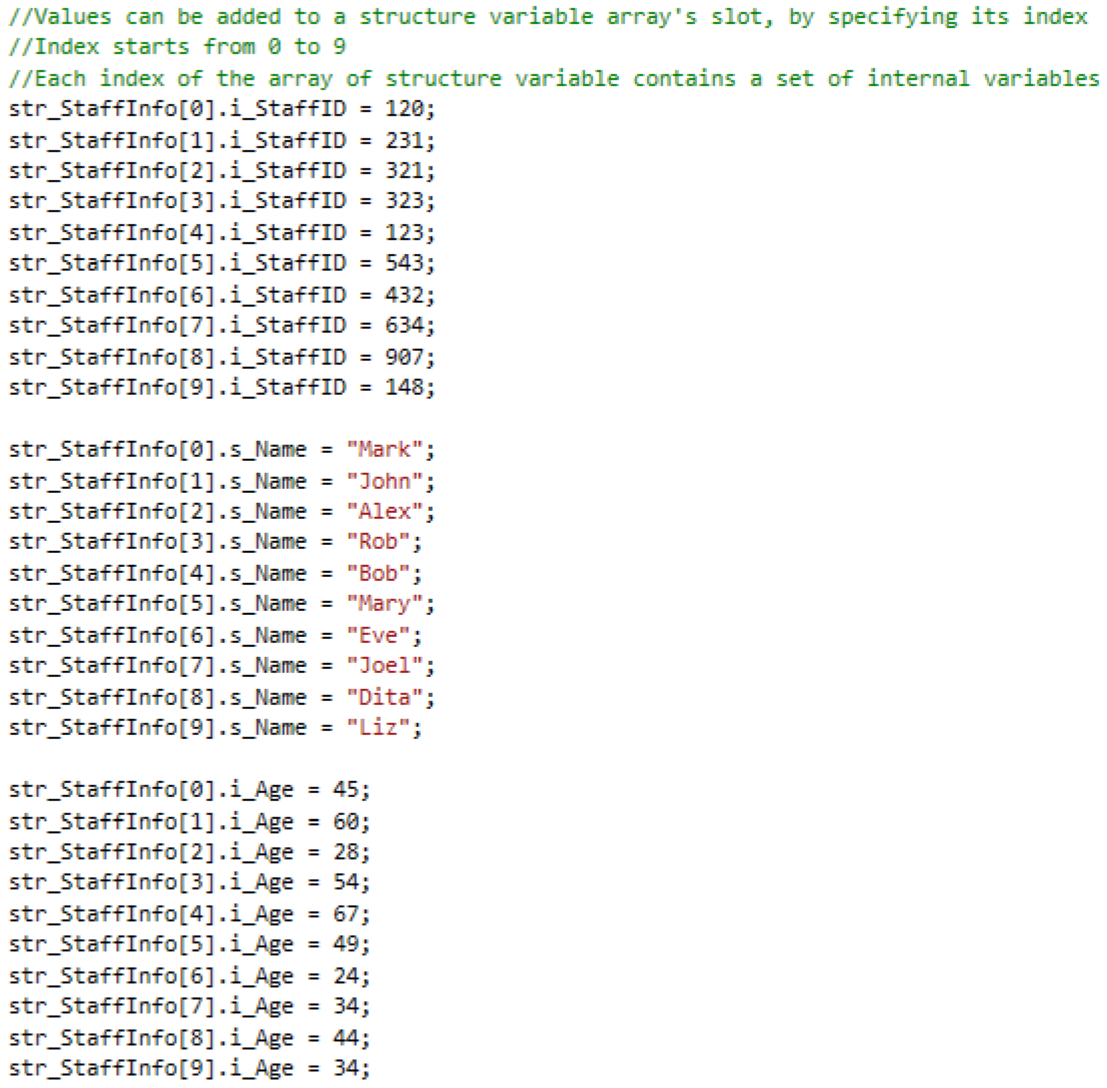


```
//Declaring an array of structure variable
strEmployee[] str_StaffInfo = new strEmployee[10];
```

In the subsequent sample codes, we will use the array of structure variable to store employee information.

Accessing values in an array of structure variable

Similar to an array of built-in variable, values stored in an array or structure variable can be accessed by specifying its index number, which starts from index zero.



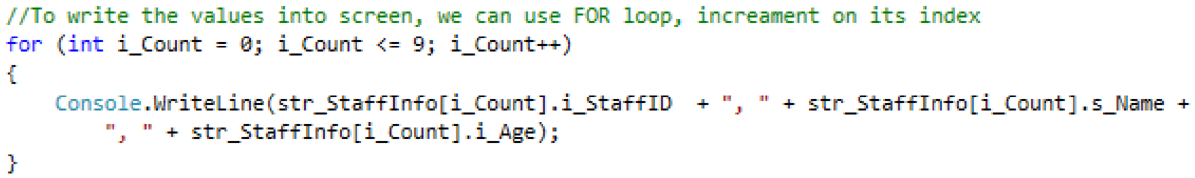
```
//Values can be added to a structure variable array's slot, by specifying its index
//Index starts from 0 to 9
//Each index of the array of structure variable contains a set of internal variables
str_StaffInfo[0].i_StaffID = 120;
str_StaffInfo[1].i_StaffID = 231;
str_StaffInfo[2].i_StaffID = 321;
str_StaffInfo[3].i_StaffID = 323;
str_StaffInfo[4].i_StaffID = 123;
str_StaffInfo[5].i_StaffID = 543;
str_StaffInfo[6].i_StaffID = 432;
str_StaffInfo[7].i_StaffID = 634;
str_StaffInfo[8].i_StaffID = 907;
str_StaffInfo[9].i_StaffID = 148;

str_StaffInfo[0].s_Name = "Mark";
str_StaffInfo[1].s_Name = "John";
str_StaffInfo[2].s_Name = "Alex";
str_StaffInfo[3].s_Name = "Rob";
str_StaffInfo[4].s_Name = "Bob";
str_StaffInfo[5].s_Name = "Mary";
str_StaffInfo[6].s_Name = "Eve";
str_StaffInfo[7].s_Name = "Joel";
str_StaffInfo[8].s_Name = "Dita";
str_StaffInfo[9].s_Name = "Liz";

str_StaffInfo[0].i_Age = 45;
str_StaffInfo[1].i_Age = 60;
str_StaffInfo[2].i_Age = 28;
str_StaffInfo[3].i_Age = 54;
str_StaffInfo[4].i_Age = 67;
str_StaffInfo[5].i_Age = 49;
str_StaffInfo[6].i_Age = 24;
str_StaffInfo[7].i_Age = 34;
str_StaffInfo[8].i_Age = 44;
str_StaffInfo[9].i_Age = 34;
```

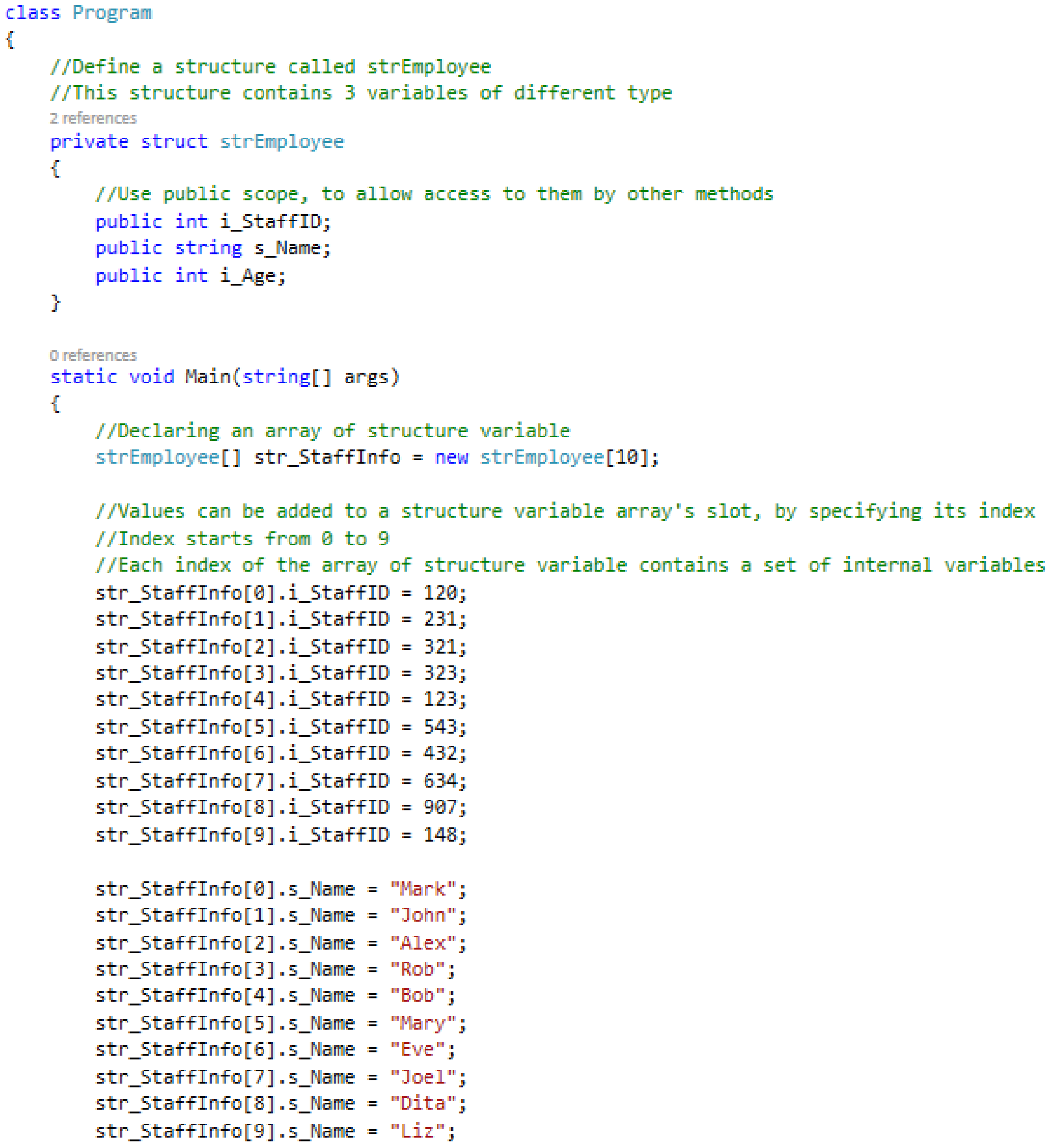
Displaying array values on screen

We can write the codes this way.



```
//To write the values into screen, we can use FOR loop, increament on its index
for (int i_Count = 0; i_Count <= 9; i_Count++)
{
    Console.WriteLine(str_StaffInfo[i_Count].i_StaffID  + ", " + str_StaffInfo[i_Count].s_Name +
        ", " + str_StaffInfo[i_Count].i_Age);
}
```

Full codes

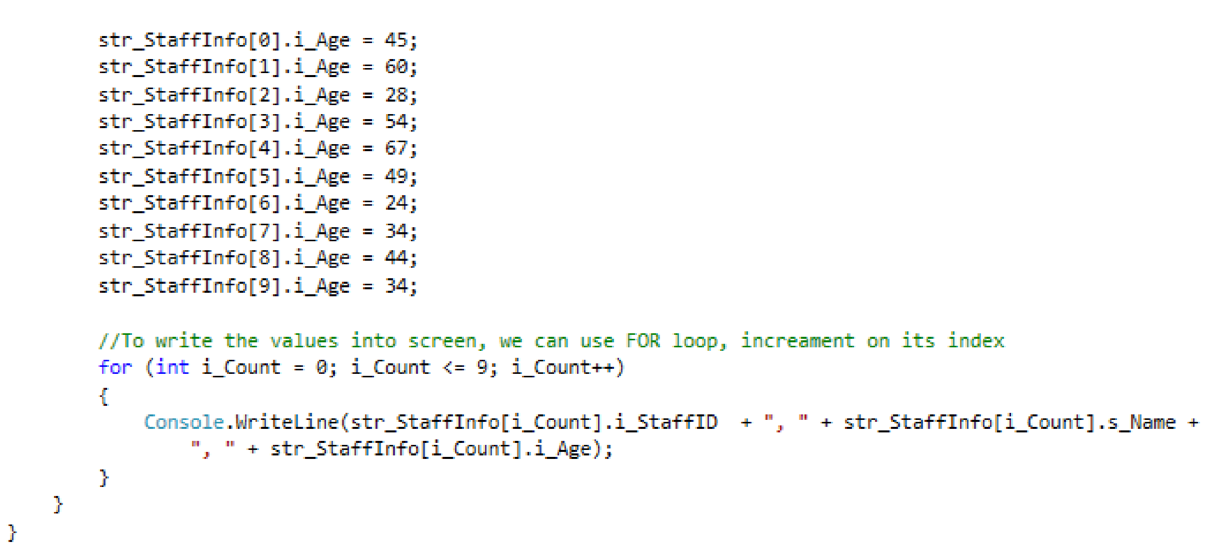


```
class Program
{
    //Define a structure called strEmployee
    //This structure contains 3 variables of different type
    2 references
    private struct strEmployee
    {
        //Use public scope, to allow access to them by other methods
        public int i_StaffID;
        public string s_Name;
        public int i_Age;
    }

    0 references
    static void Main(string[] args)
    {
        //Declaring an array of structure variable
        strEmployee[] str_StaffInfo = new strEmployee[10];

        //Values can be added to a structure variable array's slot, by specifying its index
        //Index starts from 0 to 9
        //Each index of the array of structure variable contains a set of internal variables
        str_StaffInfo[0].i_StaffID = 120;
        str_StaffInfo[1].i_StaffID = 231;
        str_StaffInfo[2].i_StaffID = 321;
        str_StaffInfo[3].i_StaffID = 323;
        str_StaffInfo[4].i_StaffID = 123;
        str_StaffInfo[5].i_StaffID = 543;
        str_StaffInfo[6].i_StaffID = 432;
        str_StaffInfo[7].i_StaffID = 634;
        str_StaffInfo[8].i_StaffID = 907;
        str_StaffInfo[9].i_StaffID = 148;

        str_StaffInfo[0].s_Name = "Mark";
        str_StaffInfo[1].s_Name = "John";
        str_StaffInfo[2].s_Name = "Alex";
        str_StaffInfo[3].s_Name = "Rob";
        str_StaffInfo[4].s_Name = "Bob";
        str_StaffInfo[5].s_Name = "Mary";
        str_StaffInfo[6].s_Name = "Eve";
        str_StaffInfo[7].s_Name = "Joel";
        str_StaffInfo[8].s_Name = "Dita";
        str_StaffInfo[9].s_Name = "Liz";
```



```
        str_StaffInfo[0].i_Age = 45;
        str_StaffInfo[1].i_Age = 60;
        str_StaffInfo[2].i_Age = 28;
        str_StaffInfo[3].i_Age = 54;
        str_StaffInfo[4].i_Age = 67;
        str_StaffInfo[5].i_Age = 49;
        str_StaffInfo[6].i_Age = 24;
        str_StaffInfo[7].i_Age = 34;
        str_StaffInfo[8].i_Age = 44;
        str_StaffInfo[9].i_Age = 34;

        //To write the values into screen, we can use FOR loop, increament on its index
        for (int i_Count = 0; i_Count <= 9; i_Count++)
        {
            Console.WriteLine(str_StaffInfo[i_Count].i_StaffID  + ", " + str_StaffInfo[i_Count].s_Name +
                ", " + str_StaffInfo[i_Count].i_Age);
        }
    }
}
```