**VALUE TYPE VS REFERENCE TYPE IN C#**

|  |  |
| --- | --- |
| **STACK**  Salary Age  5000 25  Salary Age  5000 23  Salary Age  5000 23 | **HEAP** |

e

e1

e2

struct Employee

{

public int Salary;

public int Age;

}

class Program

{

static void Main(string[] args)

{

Employee e = new Employee();

e.Salary = 5000;

e.Age = 23;

Employee e1 = e;

Employee e2 = e;

e.Age = 25;

Console.WriteLine(e1.Age);

Console.WriteLine(e2.Age);

Console.ReadLine();

}

}

|  |  |
| --- | --- |
| **STACK**  0x3453    0x3453  0x3453 | **HEAP**  Salary Age  5000 25    0x3453 |

class Employee

{

public int Salary;

public int Age;

}

class Program

{

static void Main(string[] args)

{

Employee e = new Employee();

e.Salary = 5000;

e.Age = 23;

Employee e1 = e;

Employee e2 = e;

e.Age = 25;

Console.WriteLine(e1.Age);

Console.WriteLine(e2.Age);

Console.ReadLine();

}

}

e2

e1

e

**VALUE TYPE:**

A data type is a value type if it holds a data value within its own memory space. It means variables of these data types directly contain their values.

The following data types are all of value type:

* bool
* byte
* char
* decimal
* double
* enum
* float
* int
* long
* sbyte
* short
* struct
* uint
* ulong
* ushort

**REFERENCE TYPE:**

Unlike value types, a reference type doesn't store its value directly. Instead, it stores the address where the value is being stored. In other words, a reference type contains a pointer to another memory location that holds the data.

The following data types are of reference type:

* String
* All arrays, even if their elements are value types
* Class
* object
* Delegates
* Interface
* **Difference between value type and reference types ?**

|  |  |
| --- | --- |
| **Value Type** | **Reference Type** |
| They are stored on stack memory | They are stored on heap memory |
| Contains actual value | Contains reference to a value |
| Cannot contain null values. However this can be achieved by nullable types | Can contain null values. |
| Memory is allocated at compile time | Memory is allocated at run time |

* **Diff bt stack and heap**

|  |  |
| --- | --- |
| **Stack** | **Heap** |
| Values are stored on one another like a stack. | Values are stored in random order like dumped into a huge space |
| Used for value type | Used for reference types |