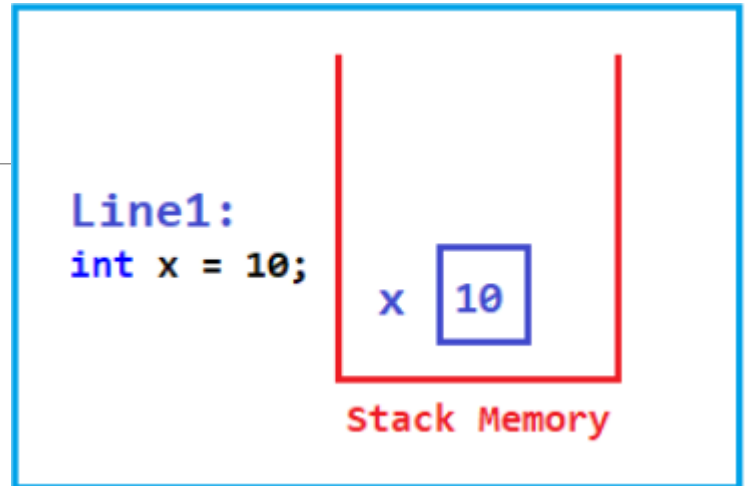
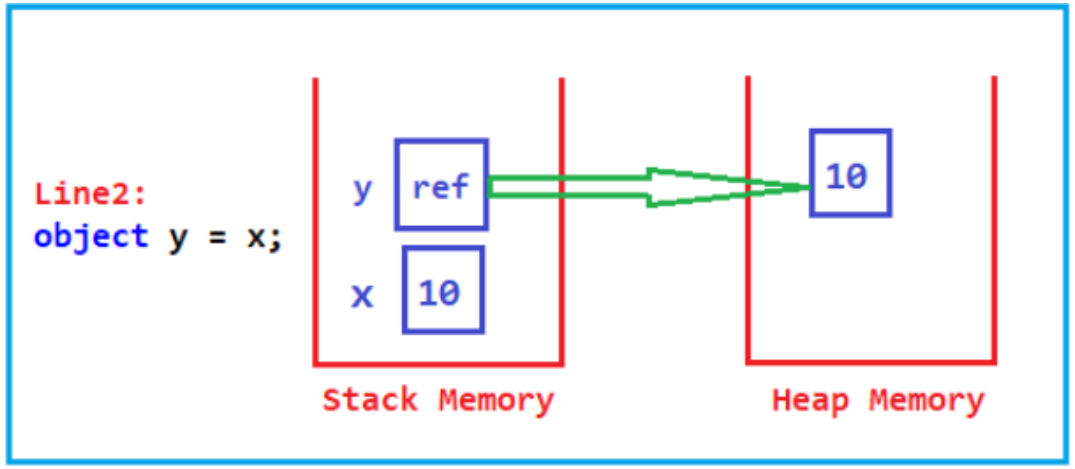
OBJECT-ORIENTED PROGRAMMING

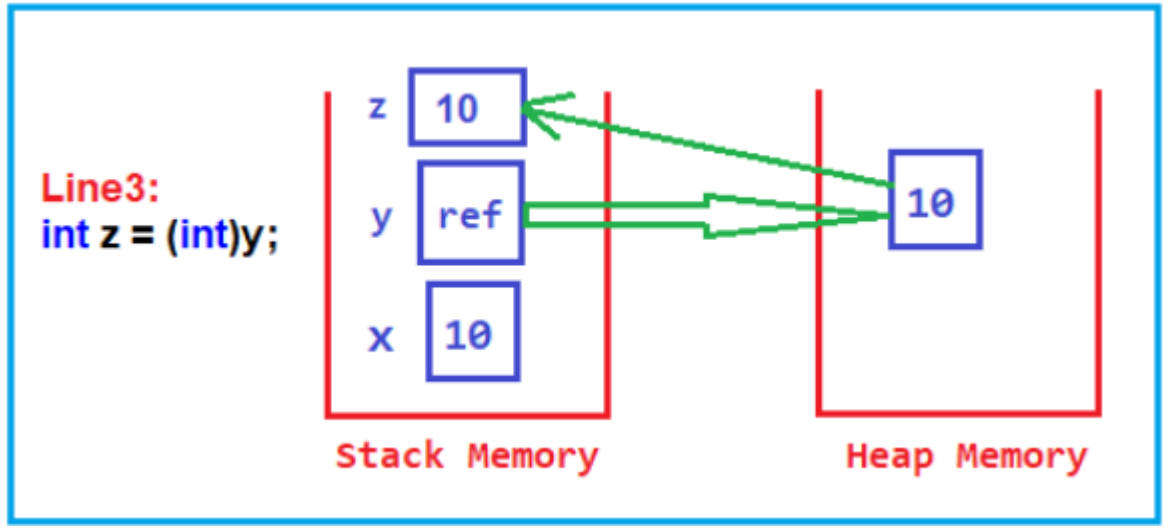
LAB6 :OOP_CONCEPTS

Boxing and Unboxing in C# with Examples

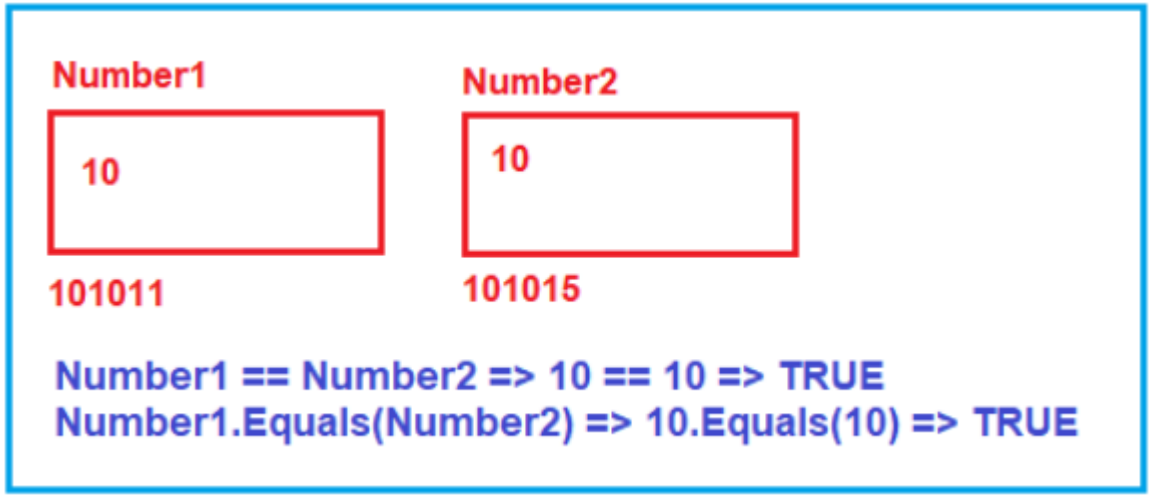
```
public void SomeMethod()
{
Line1 ⇨ int x = 10;
Line2 ⇨ object y = x; //Boxing
Line3 ⇨ int z = (int)y; //Unboxing
}
```



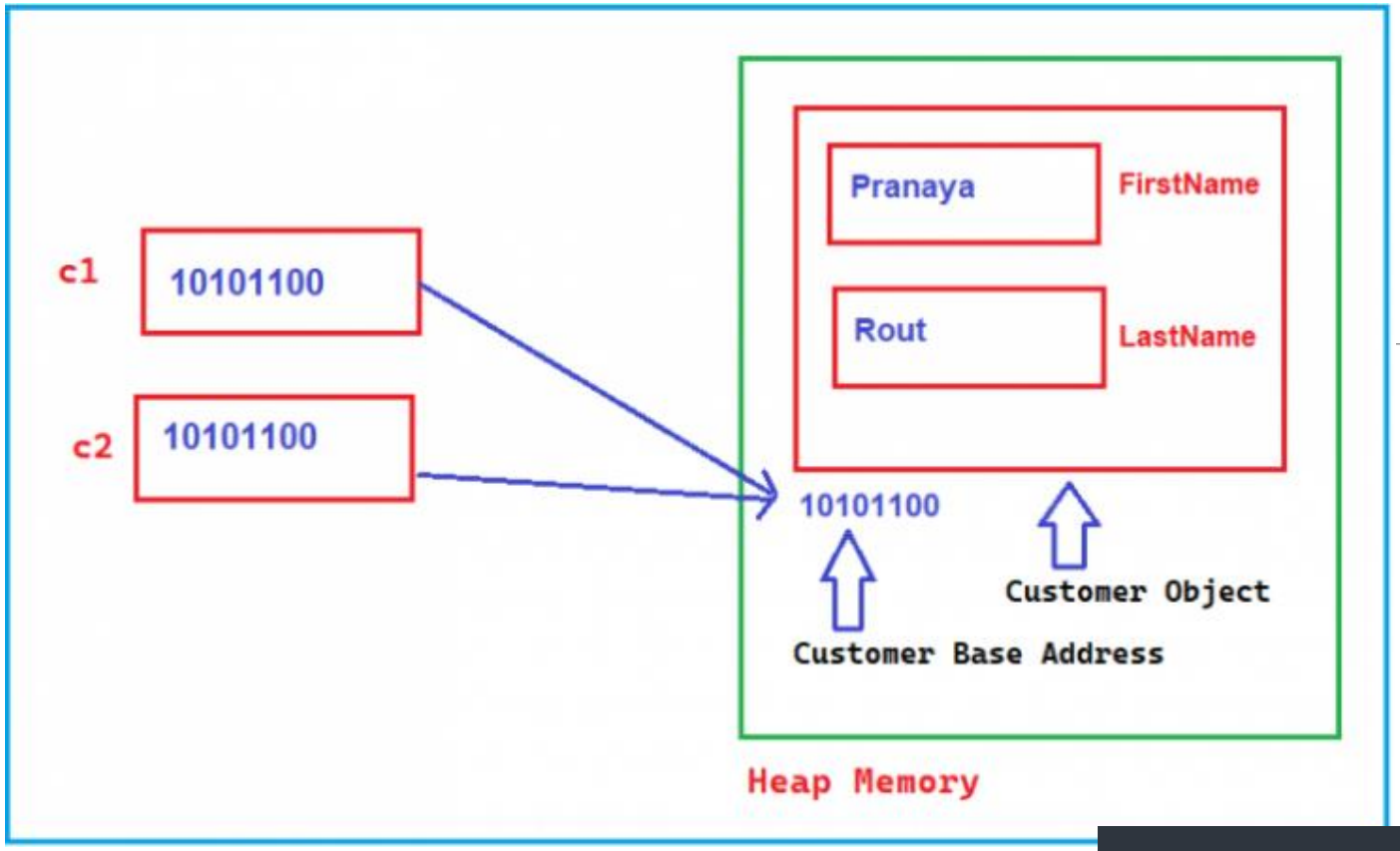




Difference Between the “==” Operator and the Equals() Method in C#:




```
    public static void Main()
    {
        Direction direction1 = Direction.East;
        Direction direction2 = Direction.East;
        Console.WriteLine(direction1 == direction2);
        Console.WriteLine(direction1.Equals(direction2));
        Console.ReadKey();
    }
}
public enum Direction
{
    East = 1,
    West = 2,
    North = 3,
    South = 4
}
```

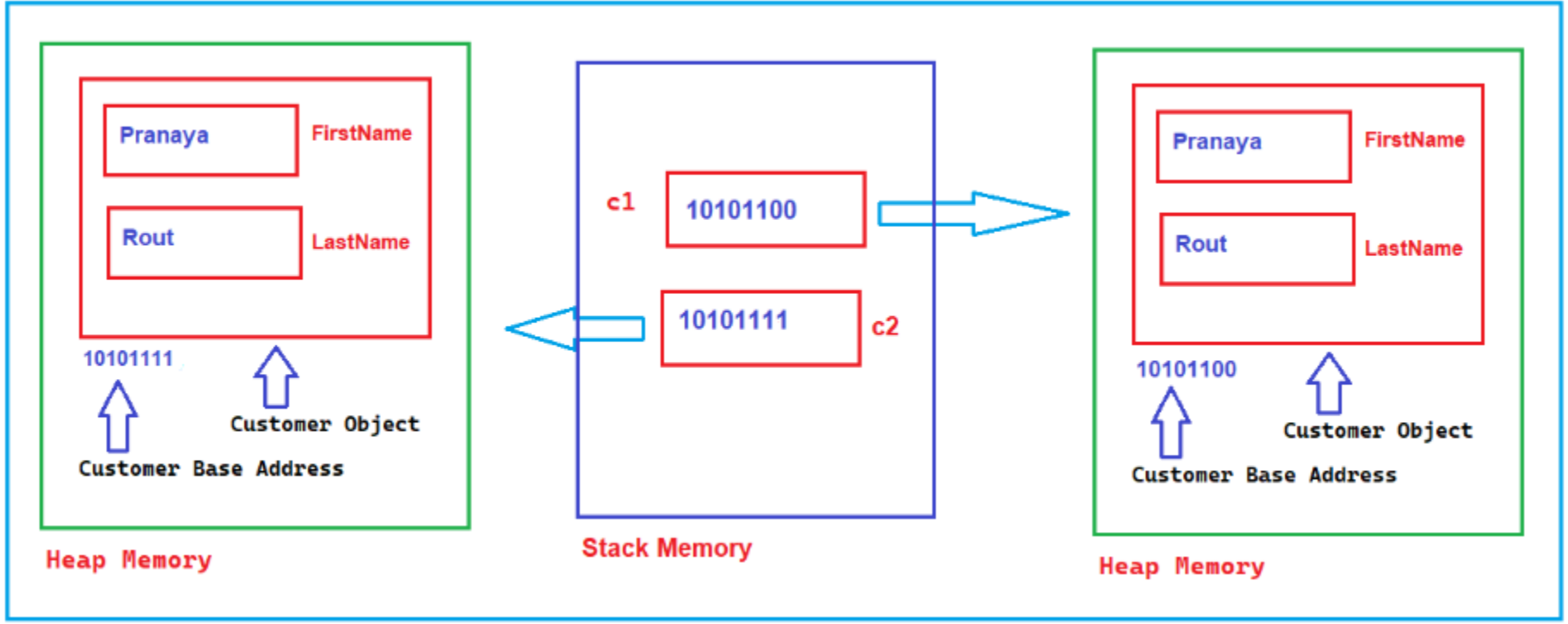



```
C1 == C2: True
C1.Equals(C2): True
```

```
Customer C1 = new Customer();
C1.FirstName = "Pranaya";
C1.LastName = "Rout";

Customer C2 = C1;

Console.WriteLine($"C1 == C2: {C1 == C2}");
Console.WriteLine($"C1.Equals(C2): {C1.Equals(C2)}");
```




```
C1 == C2: False
C1.Equals(C2): False
```

```
    Customer C1 = new Customer();
    C1.FirstName = "Pranaya";
    C1.LastName = "Rout";

    Customer C2 = new Customer();
    C2.FirstName = "Pranaya";
    C2.LastName = "Rout";

    Console.WriteLine($"C1 == C2: {C1 == C2}");
    Console.WriteLine($"C1.Equals(C2): {C1.Equals(C2)}");

    Console.ReadKey();
}
```

```
public class Customer
{
    public string FirstName { get; set; }
    public string LastName { get; set; }

    public override bool Equals(object obj)
    {
        // If the passed object is null, return False
        if (obj == null)
        {
            return false;
        }
        // If the passed object is not Customer Type, return False
        if (!(obj is Customer))
        {
            return false;
        }
        return (this.FirstName == ((Customer)obj).FirstName)
            && (this.LastName == ((Customer)obj).LastName);
    }
}
```

EXERCISE 6.1: Book Class

- This exercise is worth 2 sections.
- Write a program that first reads book information from the user.
- The details to be asked for each book include the title, the number of pages and the publication year.
- Entering an empty string as the name of the book ends the reading process.
- After this the user is asked for what is to be printed.
- If the user inputs “everything” all the details are printed: the book titles, the numbers of pages, and the publication years.
- However if the user enters the string “title” only the book titles are printed.
- If something else than “everything” or “title” is given the program should not print anything.
- Implement the class Book.
- Implement the functionality in the Main method.
- Example of how the program in Main should work.

Name: To Kill a Mockingbird

Pages: 281

Publication year: 1960

Name: A Brief History of Time

Pages: 256

Publication year: 1988

Name: Beautiful Code

Pages: 593

Publication year: 2007

Name: The Name of the Wind

Pages: 662

Publication year: 2007

Name:

What information will be printed? everything

To Kill a Mockingbird, 281 pages, 1960

A Brief History of Time, 256 pages, 1988

Beautiful Code, 593 pages, 2007

The Name of the Wind, 662 pages, 2007

Name: To Kill a Mockingbird
Pages: 281
Publication year: 1960
Name: A Brief History of Time
Pages: 256
Publication year: 1988
Name: Beautiful Code
Pages: 593
Publication year: 2007
Name: The Name of the Wind
Pages: 662
Publication year: 2007
Name:

What information will be printed? title
To Kill a Mockingbird
A Brief History of Time
Beautiful Code
The Name of the Wind

EXERCISE 6.2: Debt

- Create the class **Debt** that has double type instance variables **balance** and **interestRate**.
- The balance and the interest rate are passed to the constructor as parameters **public Debt(double initialBalance, double initialInterestRate)**.
- In addition create the methods **public void PrintBalance()** and **public void WaitOneYear()** for the class.
- The method PrintBalance prints the current balance and the WaitOneYear method grows the debt amount.
- The debt is increased by multiplying the balance by the interest rate.

الرهن العقاري
mortgage

```
public static void Main(string[] args)
{

 Debt mortgage = new Debt(120000.0, 1.01);
 mortgage.PrintBalance();

 mortgage.WaitOneYear();
 mortgage.PrintBalance();

 // Wait 20 years
 int years = 0;
 while (years < 20)
 {
   mortgage.WaitOneYear();
   years = years + 1;
 }

 mortgage.PrintBalance();
}
```

```
120000
121200
147887.0328416936
```