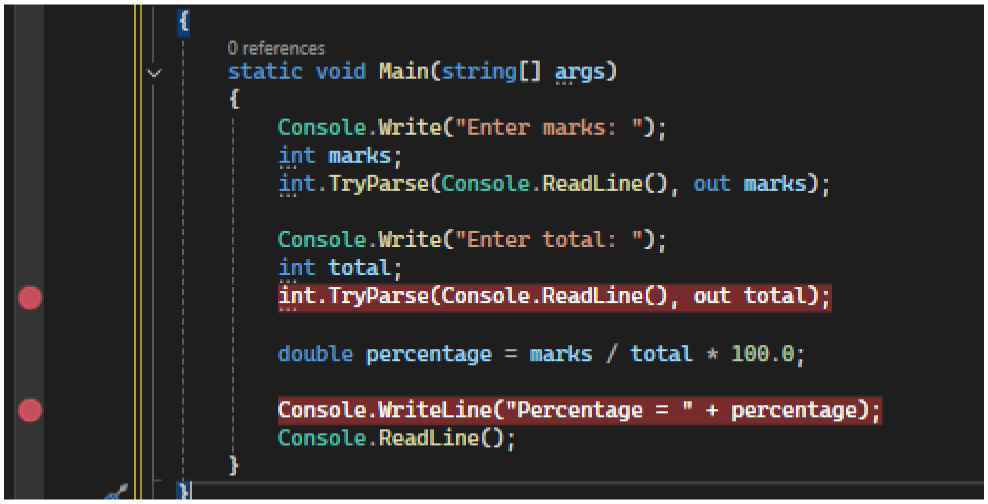
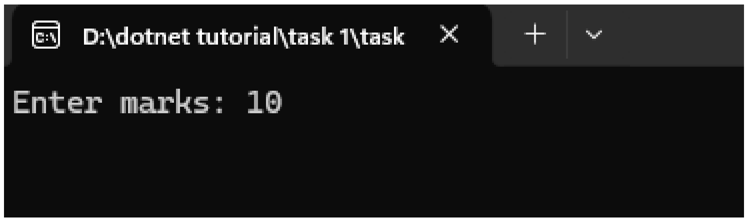
1. Task 6

Here's the code before running anything based on the question that was provided,

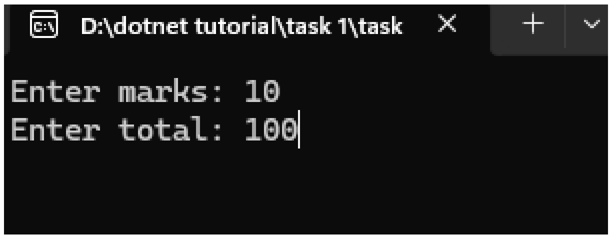


```
{
    0 references
    static void Main(string[] args)
    {
        Console.Write("Enter marks: ");
        int marks;
        int.TryParse(Console.ReadLine(), out marks);

        Console.Write("Enter total: ");
        int total;
        int.TryParse(Console.ReadLine(), out total);

        double percentage = marks / total * 100.0;

        Console.WriteLine("Percentage = " + percentage);
        Console.ReadLine();
    }
}
```

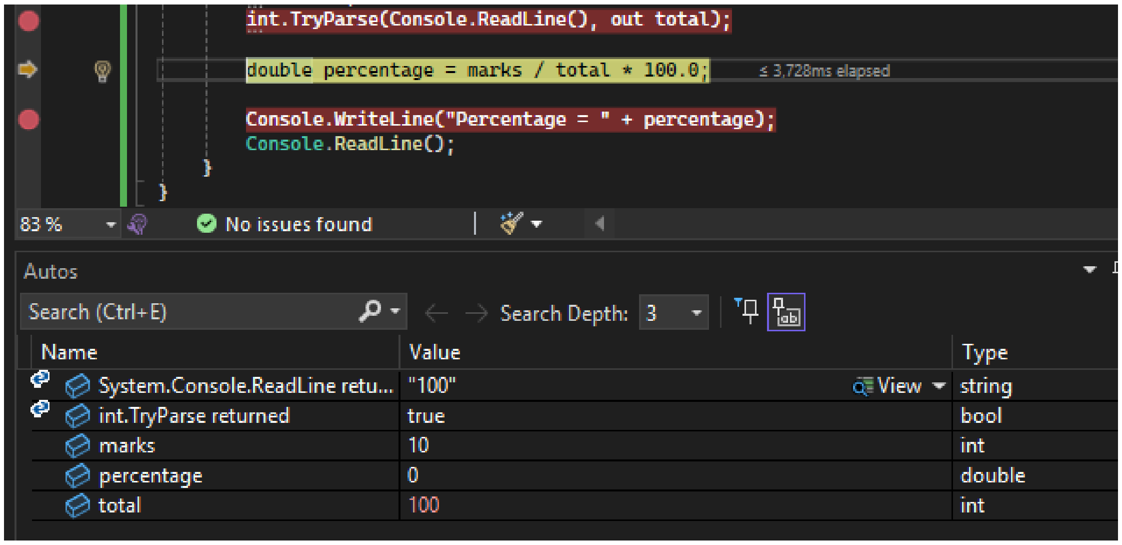
After running the program,



```
Enter marks: 10
```



```
Enter marks: 10
Enter total: 100
```



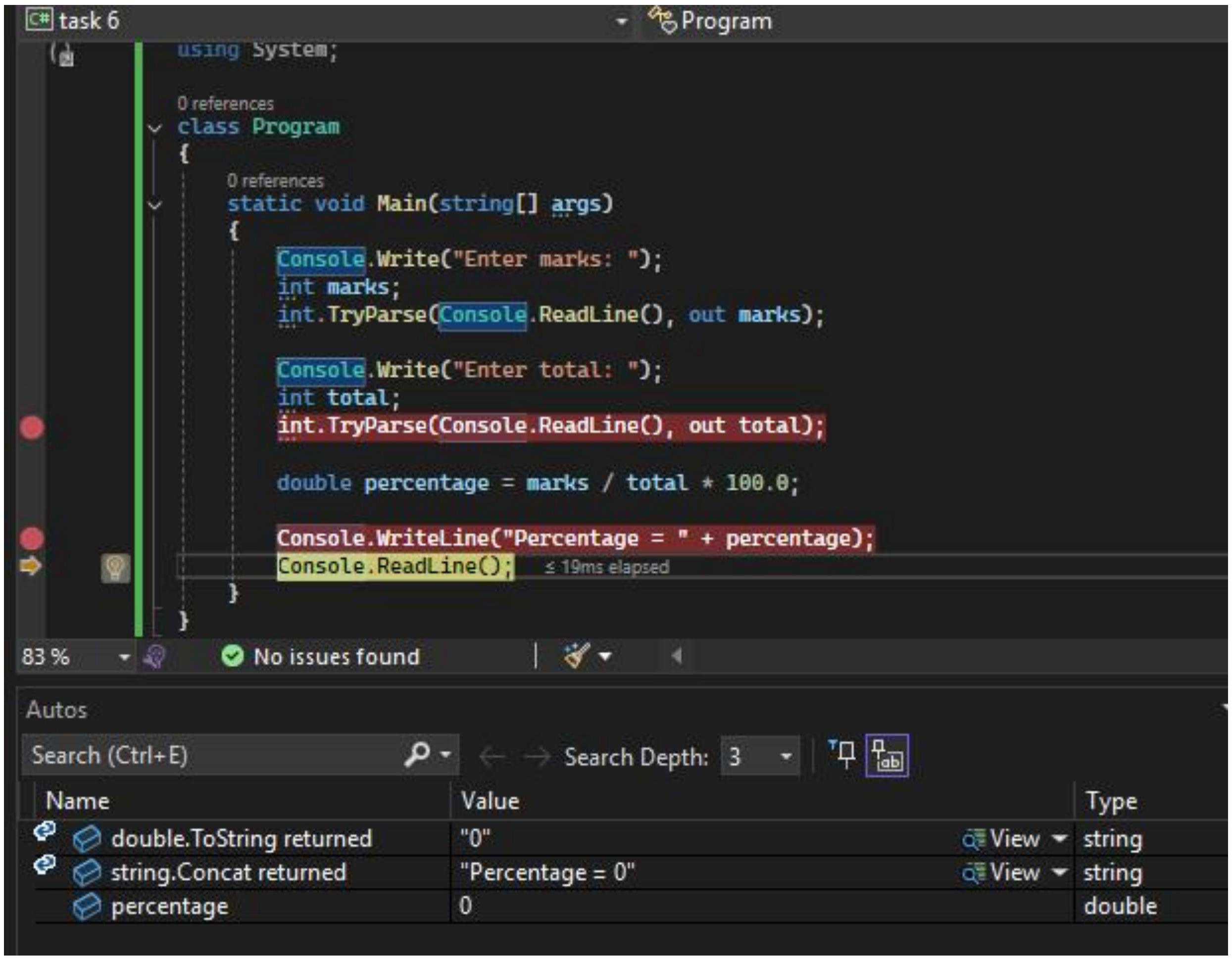
```
int.TryParse(Console.ReadLine(), out total);

double percentage = marks / total * 100.0;    ≤ 3,728ms elapsed

Console.WriteLine("Percentage = " + percentage);
Console.ReadLine();
        }
    }
```

83 %   No issues found

Autos

Search (Ctrl+E)   Search Depth: 3

Name	Value	Type
System.Console.ReadLine retu...	"100"	string
int.TryParse returned	true	bool
marks	10	int
percentage	0	double
total	100	int



task 6   Program

```
using System;

0 references
class Program
{
    0 references
    static void Main(string[] args)
    {
        Console.Write("Enter marks: ");
        int marks;
        int.TryParse(Console.ReadLine(), out marks);

        Console.Write("Enter total: ");
        int total;
        int.TryParse(Console.ReadLine(), out total);

        double percentage = marks / total * 100.0;

        Console.WriteLine("Percentage = " + percentage);
        Console.ReadLine();    ≤ 19ms elapsed
    }
}
```

83 %   No issues found

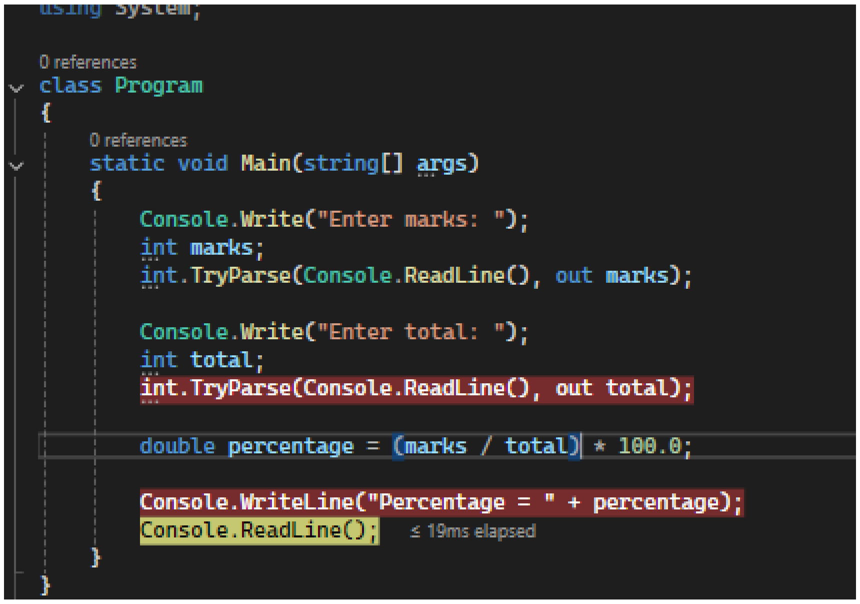
Autos

Search (Ctrl+E)   Search Depth: 3

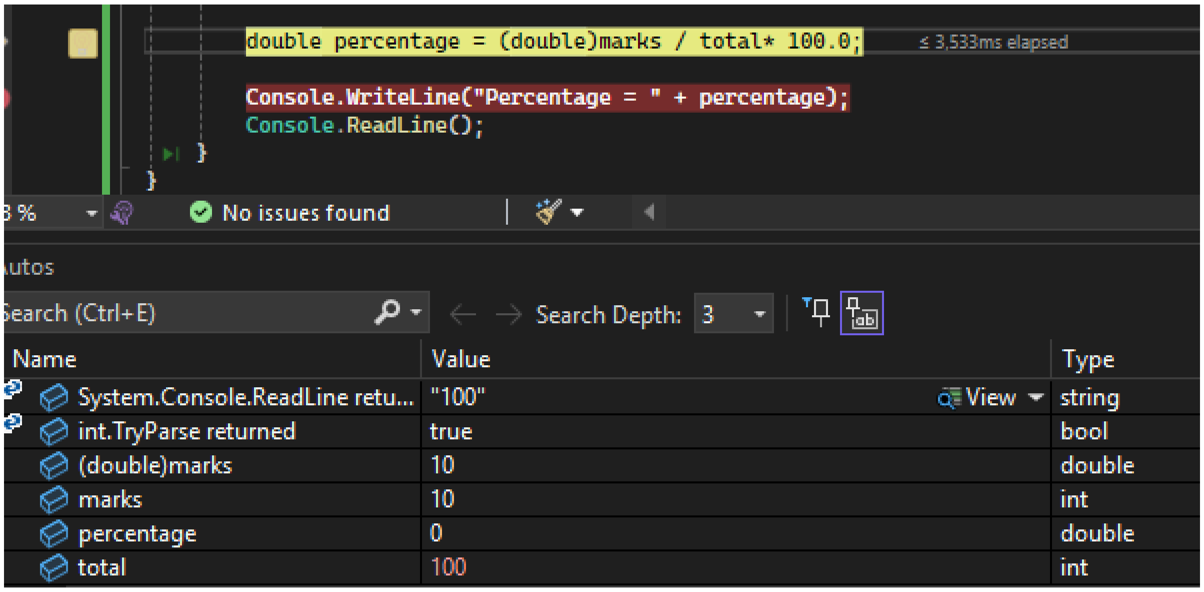
Name	Value	Type
double.ToString returned	"0"	string
string.Concat returned	"Percentage = 0"	string
percentage	0	double

The output is incorrect because the calculation uses the wrong order of operations.

The program first performs integer division (marks / total). When marks = 10 and total = 100, the division 10 / 100 results in 0 because the marks was in integer so the value didn't returned and percentage became 0

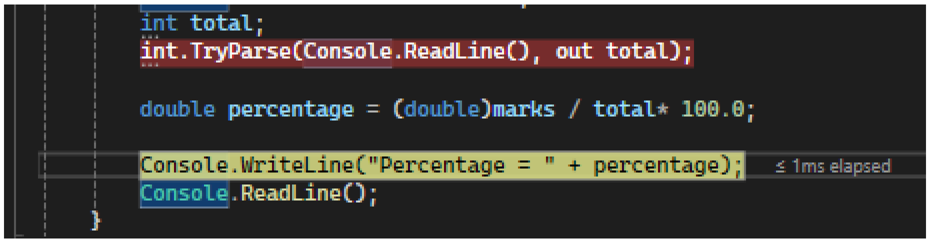


```
using System;

0 references
class Program
{
    0 references
    static void Main(string[] args)
    {
        Console.Write("Enter marks: ");
        int marks;
        int.TryParse(Console.ReadLine(), out marks);

        Console.Write("Enter total: ");
        int total;
        int.TryParse(Console.ReadLine(), out total);

        double percentage = (marks / total) * 100.0;

        Console.WriteLine("Percentage = " + percentage);
        Console.ReadLine();  ≤ 19ms elapsed
    }
}
```

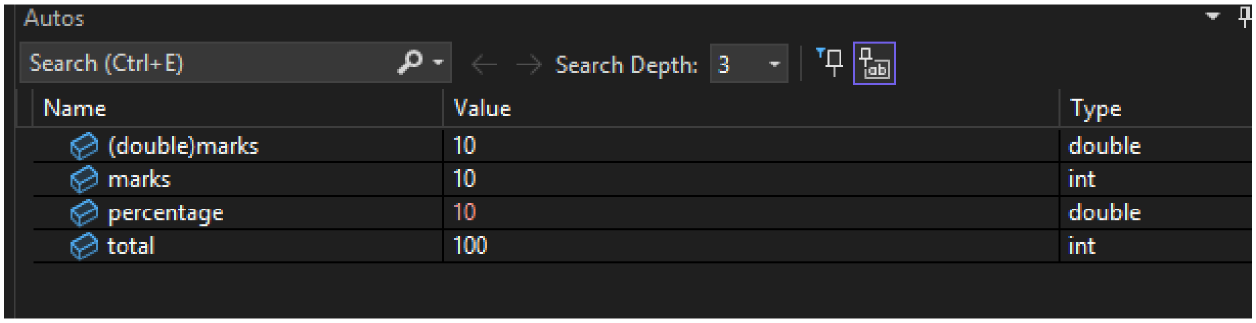


```
        double percentage = (double)marks / total* 100.0;  ≤ 3,533ms elapsed

        Console.WriteLine("Percentage = " + percentage);
        Console.ReadLine();
    }
}
```

Name	Value	Type
System.Console.ReadLine retu...	"100"	string
int.TryParse returned	true	bool
(double)marks	10	double
marks	10	int
percentage	0	double
total	100	int



```
        int total;
        int.TryParse(Console.ReadLine(), out total);

        double percentage = (double)marks / total* 100.0;

        Console.WriteLine("Percentage = " + percentage);  ≤ 1ms elapsed
        Console.ReadLine();
    }
```



Autos

Search (Ctrl+E) Search Depth: 3

Name	Value	Type
(double)marks	10	double
marks	10	int
percentage	10	double
total	100	int

But I parsed the mark to double the program correctly worked and the value retrieved successfully

2. Task 7

How constructors help in software development

Constructors are practices in a class which are called upon automatically when an object is created. They are significant in the object-oriented programming by making sure that objects begin in an acceptable and prepared state.

Donation to Object Creation.

Constructors enable us to assign preliminary values to fields when they are created. This prevents garbage or invalid data of the object.

Example: In a banking application, one can initially deposit an initial balance of 0 when creating a new Account object.

Addition to the Code Reliability.

We prevent runtime errors where uninitialised data causes runtime errors by coerced required values using parameterised constructors. In case a field is required to be supplied, the constructor will make sure that one is supplied.

Example: A Student object constructor may need a name and roll number and in that case it is impossible to make a student without this required information.

This is the contribution to maintainability.

Initialization code is stored in a single location by constructors. When the initialisation logic subsequently varies we do not have to go searching through the whole program but just update the constructor. This simplifies the process of updating and comprehending the code.

Three Real-World Use Cases

Game Development

In the case of a game where a new Player object is being created, the default values such as health = 100, level = 1, and position on the map have been assigned in the constructor. This means that all the players get the game right.

E-commerce Application

Cart item object is generated when a user orders an item in the cart. The name of the product, price and quantity are automatically entered by the constructor and errors such as adding a product with no price are avoided.

Hospital Management System

Upon creation of a new Patient record, the constructor will give it a unique patient ID and the date of admission and a blank list of medical history will be created. This maintains all records of patients complete and consistent.

3. OOP Principle

Classes and Objects

A class is a blue print or design, whereas an object is the actual object that was made based on the blue print. As an example, the design of the smartphone is the class and the phone in your hand is an object. In the same manner, the recipe of a cake is the class and the real cake that is baked in the kitchen is an object.

Encapsulation

Encapsulation refers to the concealment of the internal information and only a right way allows it to be accessed. Similar to the fact that only you, and no one else, know your bank balance, and can only see or modify it at the ATM or in the app, or that the bitter powder in a medicine capsule remains invisible, or just like that capsule, encapsulation hides data and enables controlled access.

Inheritance

Inheritance refers to the automatic acquisition of features of a parent by a child. That is, each child without any effort will inherit eye color, height or facial features of parents. Similarly, a tiger is an animal and hence it inherently acquires the capability of walking and feeding as all animals, and has its own stripes and roar.

Abstraction

Abstraction refers to concealing the complex information and presenting the simple and needy information. When you turn on a remotely controlled fan, you do not have to concern yourself with the mechanism of how the motor spins and how the electricity circulates inside, you simply press the button and it is on. Equally, as the case is with Google Maps, you are not looking at the calculations of thousands of calculations that are taking place under the surface to determine the quickest route.

Polymorphism

The concept of polymorphism implies that the same action acts differently when it is performed by different people. Indicatively, when a teacher asks the children to raise their hands, all children will raise their hands in different ways, some are fast and others are slow. Or when you press the horn button, a car beeps, a truck makes a big noise, a bike makes a different noise, same thing and different consequence.

These four concepts, encapsulation, inheritance, abstraction and polymorphism, are applied in nearly all the apps such as Instagram, Paytm, Ola, WhatsApp and games to ensure that they are secure, simple to upgrade and operate efficiently.