

# What is Infinite Number?

Infinite number is a plugin that allow users to do math functions with strings. When compared with max value `Int` can take, which is a only 10 digit number (2,147,483,647) or compared with `long` which is the longest default variable in C# and have a range of 20 digit, infinite number can work with 2.1 billion digits long numbers and removes the limitation of the small Max/Min values of the variables.

In Summary, it let you work with really long numbers.

## Starting:

Importing the asset from the asset store is enough, there is no extra steps needed.

## Usage:

On any script simply adding “`using InfiniteNumber;`” will be enough.

```
using InfiniteNumber;
```

After that any function can be called with “`Infinity.FunctionName()`”. Here functionName needs to be replaced with the desired function.

For example:

```
public string firstNumber = "10";  
public string secondNumber = "20";  
public string result;  
  
private void Start()  
{  
    ...  
    result = Infinity.Addition(firstNumber, secondNumber);  
}
```

Which will give the result of 30.

## Function Names And How to Use Them:

For now there are 11 different functions but with combination of the existing functions different type of algorithms can be created.

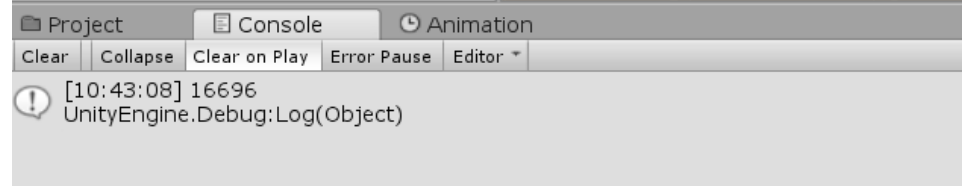
Replace the “numberOne” and “numberTwo” with your own variables.

Addition: Add two numbers together.

### Infinity.Addition(numberOne, numberTwo)

```
public string numberOne = "-970";
public string numberTwo = "17666";

public void Start()
{
    string result = Infinity.Addition(numberOne, numberTwo);
    Debug.Log(result);
}
```

The screenshot shows the Unity console with the 'Console' tab selected. It displays a log message: '[10:43:08] 16696 UnityEngine.Debug:Log(Object)'. The console interface includes buttons for 'Clear', 'Collapse', 'Clear on Play', 'Error Pause', and an 'Editor' dropdown menu.

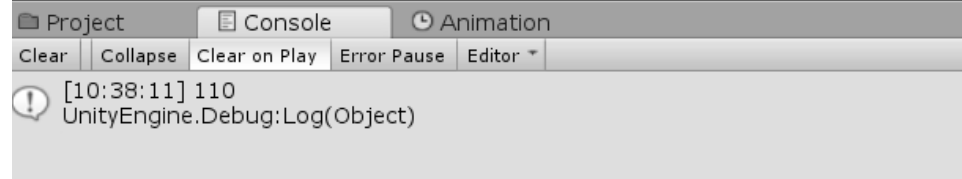
$$-970 + 17666 = 16696$$

Subtraction: Subtract second number from the first one.

### Infinity.Subtraction(numberOne, numberTwo)

```
public string numberOne = "130";
public string numberTwo = "20";

public void Start()
{
    string result = Infinity.Subtraction(numberOne, numberTwo);
    Debug.Log(result);
}
```

The screenshot shows the Unity console with the 'Console' tab selected. It displays a log message: '[10:38:11] 110 UnityEngine.Debug:Log(Object)'. The console interface includes buttons for 'Clear', 'Collapse', 'Clear on Play', 'Error Pause', and an 'Editor' dropdown menu.

$$130 - 20 = 110$$

Multiplication: Multiply the numbers with each other.

### Infinity.Multiplication(numberOne, numberTwo)

```
public string numberOne = "7854784537834578354789354789354789345789345784564569534";
public string numberTwo = "9303453465873645892375023742307429534856340583049572380462395823472398562389742349";

public void Start()
{
    string result = Infinity.Multiplication(numberOne, numberTwo);
    Debug.Log(result);
}
```

7854784537834578354789354789354789345789345784564569534

\*

9303453465873645892375023742307429534856340583049572380462395824  
72398562389742349

=

7307662243220783183864905080774825529894787726039143283329051461  
1779238374072002176170825106312792386944131899263489640535451947  
600354995366

Division: Divide the first number to the second number.

### Infinity.Division(numberOne, numberTwo)

```
public string numberOne = "767892";
public string numberTwo = "12";

public void Start()
{
    string result = Infinity.Division(numberOne, numberTwo);
    Debug.Log(result);
}
```

Project Console Animation  
Clear Collapse Clear on Play Error Pause Editor ▾  
[00:47:05] 63991  
UnityEngine.Debug:Log(Object)

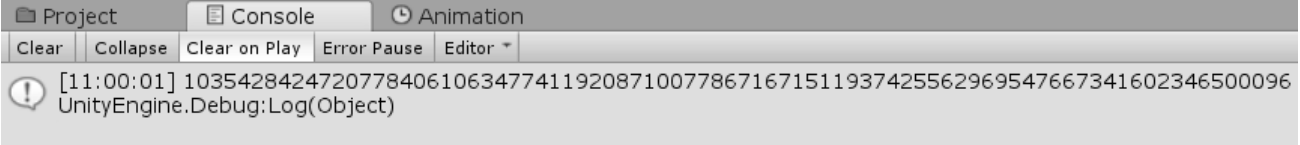
767892 / 12 = 63991

Power: Take the (second number)th power of the first number.

### **Infinity.Power(numberOne, numberTwo)**

```
public string numberOne = "178";
public string numberTwo = "36";

public void Start()
{
    string result = Infinity.Power(numberOne, numberTwo);
    Debug.Log(result);
}
```



The screenshot shows the Unity Console with the 'Console' tab selected. It displays a log message from 'UnityEngine.Debug:Log(Object)' at [11:00:01] with a very long string of digits representing the result of 178 to the power of 36.

178<sup>36</sup> = 1035428424720778406106347741192087100778671671511937425562969547667341602346500096

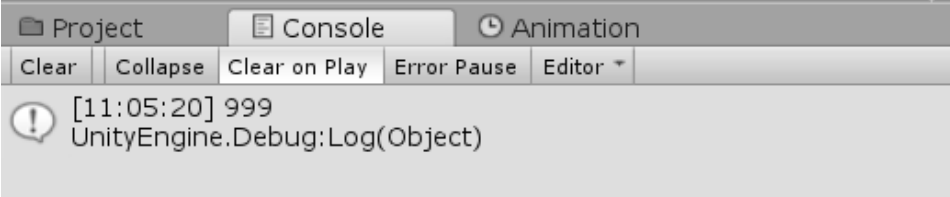
**Warning:** Taking powers of a number can take some time if the power is too big, saying 25 ^ 100000 means you want your device to make 100.000 multiplication operation and this can take some time depending on the hardware of the device. On the other hand, saying 100000 ^ 25 means your device only need to do 25 multiplication operation which should be done almost instantly.

AbsoluteValue: Take the absolute value of the given number.

### **Infinity.AbsoluteValue(numberOne)**

```
public string numberOne = "-999";

public void Start()
{
    string result = Infinity.AbsoluteValue(numberOne);
    Debug.Log(result);
}
```



The screenshot shows the Unity Console with the 'Console' tab selected. It displays a log message from 'UnityEngine.Debug:Log(Object)' at [11:05:20] with the value '999', which is the absolute value of the input '-999'.

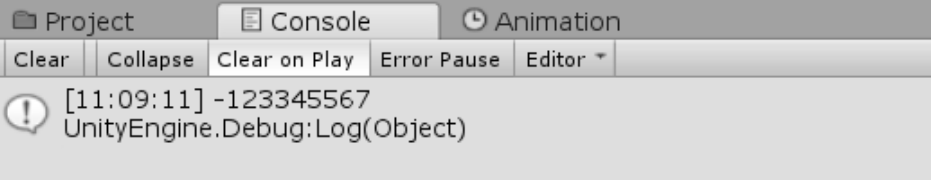
|-999| = 999

Cleaning: Make sure the variable is a number.

**Infinity.Cleaning(numberOne)**

```
public string numberOne = "-123abc345***:)567,";

public void Start()
{
    string result = Infinity.Cleaning(numberOne);
    Debug.Log(result);
}
```



The screenshot shows the Unity Console with the 'Console' tab selected. The log entry is: [11:09:11] -123345567  
UnityEngine.Debug:Log(Object)

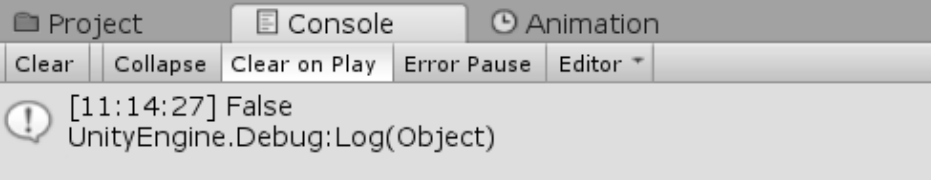
-123abc345\*\*\*:)567 = -123345567

EvenNumber: Check if the number is a even number.

**Infinity.EvenNumber(numberOne)**

```
public string numberOne = "77";

public void Start()
{
    bool result = Infinity.EvenNumber(numberOne);
    Debug.Log(result);
}
```



The screenshot shows the Unity Console with the 'Console' tab selected. The log entry is: [11:14:27] False  
UnityEngine.Debug:Log(Object)

77 is not an even number

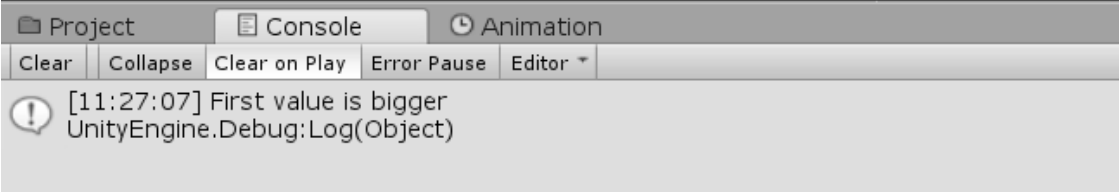
WhichValueIsBigger: Check which value is bigger (ignore negativity).

### **Infinity.WhichValueIsBigger(numberOne, numberTwo)**

“This function returns to “1” if first value is bigger, returns to “0” if values are equal and returns to “-1” if second value is bigger.”

```
public string numberOne = "-400";
public string numberTwo = "300";

public void Start()
{
    string result = Infinity.WhichValueIsBigger(numberOne, numberTwo);
    if (result == "1") Debug.Log("First value is bigger");
    else if (result == "0") Debug.Log("Values are equal");
    else if (result == "-1") Debug.Log("Second value is bigger");
}
```



400 have bigger numerical value than 300

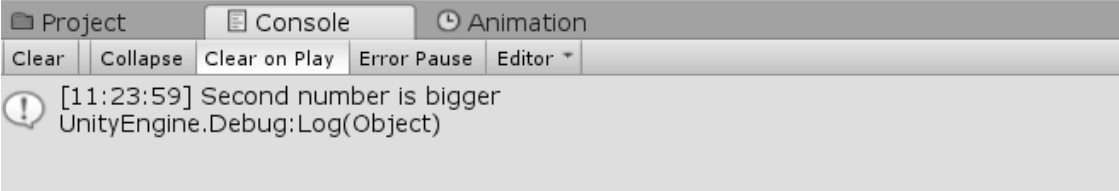
WhichNumberIsBigger: Check witch number is bigger (Does not ignore negativity).

### **Infinity.WhichNumberIsBigger(numberOne, numberTwo)**

“This function returns to “1” if first number is bigger, returns to “0” if numbers are equal and returns to “-1” if second number is bigger.”

```
public string numberOne = "-400";
public string numberTwo = "300";

public void Start()
{
    string result = Infinity.WhichNumberIsBigger(numberOne, numberTwo);
    if (result == "1") Debug.Log("First number is bigger");
    else if (result == "0") Debug.Log("Numbers are equal");
    else if (result == "-1") Debug.Log("Second number is bigger");
}
```



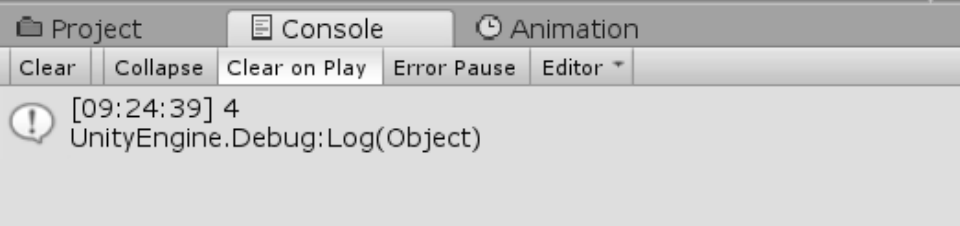
300 is bigger number than -400

CountLength: Returns Length of the number.

**Infinity.CountLength(numberOne)**

```
public string numberOne = "9987";

private void Start()
{
    string result = Infinity.CountLength(numberOne);
}
```

The screenshot shows a Unity console window with tabs for Project, Console, and Animation. The Console tab is active, displaying a log message: [09:24:39] 4 UnityEngine.Debug:Log(Object). The console window has buttons for Clear, Collapse, Clear on Play, Error Pause, and Editor.

Length of the number "9987" is 4

Difference between this function and C# string.Length is **CountLength** return to string and **string.Length** return to int, also this function does not count minus or plus signs or the unnecessary zeros in the beging, for example **CountLength** will return to 3 for the both number "-916" **string.Length** will return to 4.

**Important Note:** Unity console have a limit of how many characters it can show (15.500).

Numbers longer than 15.500 digits will not be shown in unity console completely (only first 15.500 digits will be visible) but there is no problem with numbers itself and it is safe to work with longer numbers. Also, long numbers can be fully seen in different ways. (With the usage of unity UI, with writing the number to txt, etc)