

# Ejercicios sobre manejo de excepciones en C#

Iris Pérez Aparicio

2º DAM

Campus FP Emprende Humanes

## Ejercicio 1: conversión segura de números.

**Objetivo:** Usar `try/catch` y `TryParse`.

**Descripción:** pedir un número, convertirlo a entero de forma segura usando `TryParse` y manejar números negativos con `try/catch`.

### Entrada de usuario:

Ingresa un número: `abc`

Ingresa un número: `-5`

Ingresa un número: `10`

### Salida de consola:

Valor no válido

Error: El número no puede ser negativo

Número aceptado: 10

```
using System;

class Program
{
    static void Main()
    {
        int num;

        Console.Write("Ingresa un número: ");
        string entrada = Console.ReadLine();

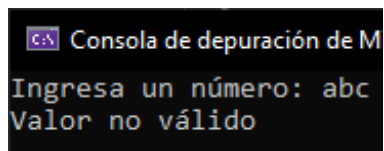
        try
        {
            // Convierto el número a entero con TryParse, si no se puede se muestra "Valor no válido"
            if (!int.TryParse(entrada, out num))
            {
                Console.WriteLine("Valor no válido");
            }
        }
    }
}
```

```

        else if (num < 0)
        {
            // Lanzo una excepción personalizada si es negativo
            throw new Exception("El número no puede ser negativo");
        }
        else
        {
            // Si es válido y positivo o cero, salir del ciclo
            Console.WriteLine("Número aceptado: " + num);
        }
    }
}
catch (Exception e)
{
    Console.WriteLine("Error: " + e.Message);
}
}
}

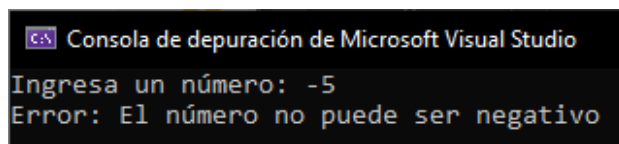
```

## EJECUCIÓN:



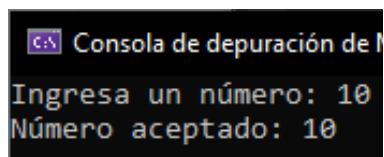
Consola de depuración de Microsoft Visual Studio

Ingresa un número: abc  
Valor no válido



Consola de depuración de Microsoft Visual Studio

Ingresa un número: -5  
Error: El número no puede ser negativo



Consola de depuración de Microsoft Visual Studio

Ingresa un número: 10  
Número aceptado: 10

## Ejercicio 2: acceso seguro a un diccionario.

**Objetivo:** Usar `Dictionary.TryGetValue`.

**Descripción:** crear un diccionario con nombres de alumnos y edades, y buscar un nombre de forma segura usando `TryGetValue`.

**Diccionario:**

- Ana → 25
- Luis → 30

### Entrada de usuario:

Ingresa nombre del alumno: Ana

Ingresa nombre del alumno: Pedro

### Salida de consola:

Edad: 25

Alumno no encontrado

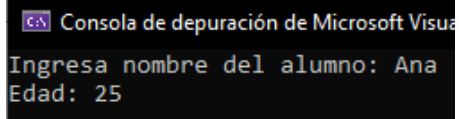
```
using System;

class Program
{
    static void Main()
    {
        // Creo el diccionario de alumnos y edades
        var alumnos = new Dictionary<string, int>
        {
            { "Ana", 25 },
            { "Luis", 30 }
        };

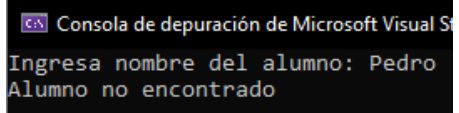
        Console.Write("Ingresa nombre del alumno: ");
        string nombre = Console.ReadLine();

        // Busco el nombre en el diccionario de forma segura y muestro la edad del alumno
        if (alumnos.TryGetValue(nombre, out int edad))
        {
            Console.WriteLine("Edad: " + edad);
        }
        else
        {
            Console.WriteLine("Alumno no encontrado");
        }
    }
}
```

### EJECUCIÓN:



```
Consola de depuración de Microsoft Visual Studio
Ingresa nombre del alumno: Ana
Edad: 25
```



```
Consola de depuración de Microsoft Visual Studio
Ingresa nombre del alumno: Pedro
Alumno no encontrado
```

## Ejercicio 3: acceso seguro a listas

**Objetivo:** Usar `ElementAtOrDefault` y `TryParse`.

**Descripción:** manejar índices de listas de forma segura usando `ElementAtOrDefault` y `TryParse`.

**Lista de palabras:** A, B, C, D, E

**Entrada de usuario:**

Ingresar índice: 2

Ingresar índice: 10

Ingresar índice: abc

**Salida de consola:**

Palabra: "C"

Índice fuera de rango

Índice no válido

```
using System;

class Program
{
    static void Main()
    {
        List<string> palabras = new List<string> { "A", "B", "C", "D", "E" };

        Console.Write("Ingresar índice: ");
        string entrada = Console.ReadLine();

        if (!int.TryParse(entrada, out int indice))
        {
            Console.WriteLine("Índice no válido");
        }
        else
        {
            string palabra = palabras.ElementAtOrDefault(indice);
            if (palabra == null)
            {
                Console.WriteLine("Índice fuera de rango");
            }
            else
            {
                Console.WriteLine($"Palabra: \"{palabra}\"");
            }
        }
    }
}
```

## EJECUCIÓN:

```
C:\> Consola de depuración de M...
Ingresa índice: 2
Palabra: "C"
```

```
C:\> Consola de depuración de M...
Ingresa índice: 10
Índice fuera de rango
```

```
C:\> Consola de depuración de M...
Ingresa índice: abc
Índice no válido
```

## Ejercicio 4: manejo mixto

**Objetivo:** Combinar `try/catch` y métodos Try.

**Descripción:** combinar acceso seguro a diccionarios y conversión de tipos con manejo de errores.

### Diccionario de productos y precios:

- Laptop → 1500
- Mouse → 25
- Teclado → 45

### Entrada de usuario:

Ingresa producto: Laptop, cantidad: 2

Ingresa producto: Tablet

Ingresa producto: Mouse, cantidad: abc

### Salida de consola:

Total: 3000

Producto no encontrado

Cantidad no válida

```

using System;

class Program
{
    static void Main()
    {
        // Creo un diccionario de productos y precios
        var productos = new Dictionary<string, double>(StringComparer.OrdinalIgnoreCase)
        {
            { "Laptop", 1500 },
            { "Mouse", 25 },
            { "Teclado", 45 }
        };

        Console.Write("Ingresa producto: ");
        string producto = Console.ReadLine();

        // Busco el producto en el diccionario de forma segura y muestro el precio
        if (!productos.TryGetValue(producto, out double precio))
        {
            Console.WriteLine("Producto no encontrado");
            return;
        }

        Console.Write("Ingresa cantidad: ");
        try
        {
            int cantidad = int.Parse(Console.ReadLine());
            double total = precio * cantidad;
            Console.WriteLine($"Total: {total}");
        }
        catch (Exception e) {
            Console.WriteLine("Cantidad no válida");
        }
    }
}

```

## EJECUCIÓN:

```

C:\> Consola de depuración de Micro
Ingresa producto: Laptop
Ingresa cantidad: 2
Total: 3000

```

```

C:\> Consola de depuración de Micro
Ingresa producto: Tablet
Producto no encontrado

```

```

C:\> Consola de depuración de Micro
Ingresa producto: Mouse
Ingresa cantidad: abc
Cantidad no válida

```