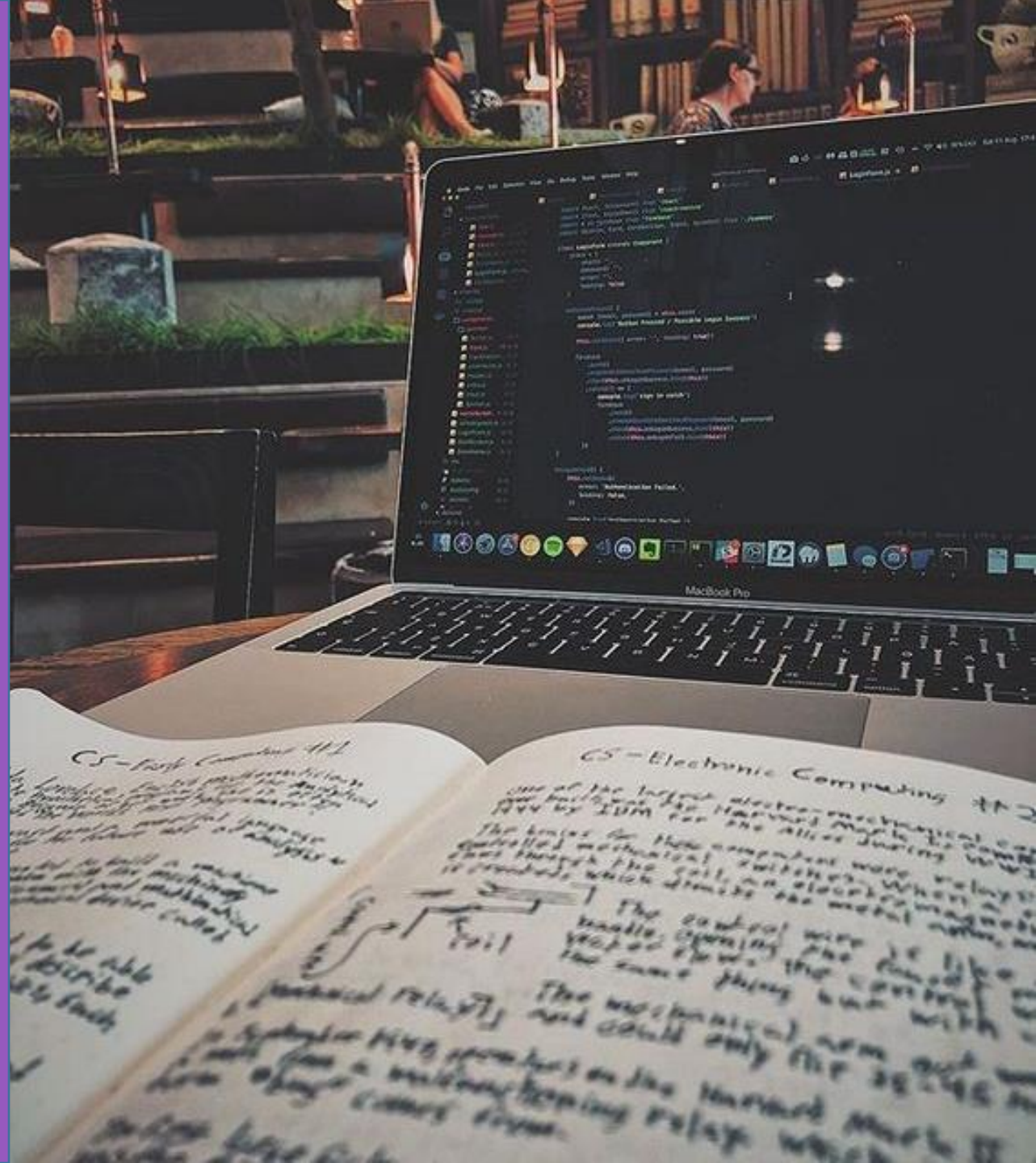
Clase Nro 1

Manejo de Excepciones

- Introducción a errores: Errores en el software
- Ciclo de vida de una Aplicación
- Tiempo de vida de una aplicación
- Tipos de errores
- Definición de Excepciones
- Tipos de Excepciones
- Gestión de Excepciones
- Try-Catch
- Try-Catch-Finally
- Capturar una excepción.
- Excepciones derivadas.
- Filtros de Excepción.
- Lanzar una excepción. (instrucción Throw)
- Crear una excepción.



Contexto del cursado de taller

Taller I

Manejos de archivos

Noción de clase, como estructura de datos

Reservas dinámicas

Usando primero lenguaje c

Después lenguaje c#

Taller II

Crear una aplicación web con modelo MVC Utilizando C# y POO

Contexto del cursado de taller

T.1 Manejo de Excepciones

T.2 Logs de Aplicaciones

+

Practicas con el IDE

Puntos de interrupción

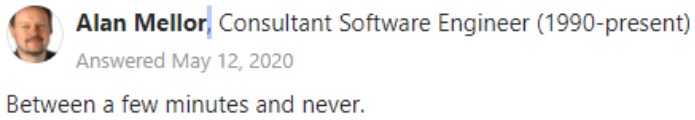
Ejercicios de debugging

Practicas sobre el manejo y la gestión de errores en una aplicación

Errores en el software

¿Cuanto tiempo te toma corregir un error en un programa?

Entre algunos minutos y nunca.



Alan Mellor, Consultant Software Engineer (1990-present)
Answered May 12, 2020

Between a few minutes and never.

https://www.quora.com/How-long-does-it-take-you-to-fix-a-program-bug

Los errores en el Software

Desde la economía y lo humano

Solo en el año 2016... (*)

4,4

Billones Personas

1,1

Trillon de dólares en recursos

315 años

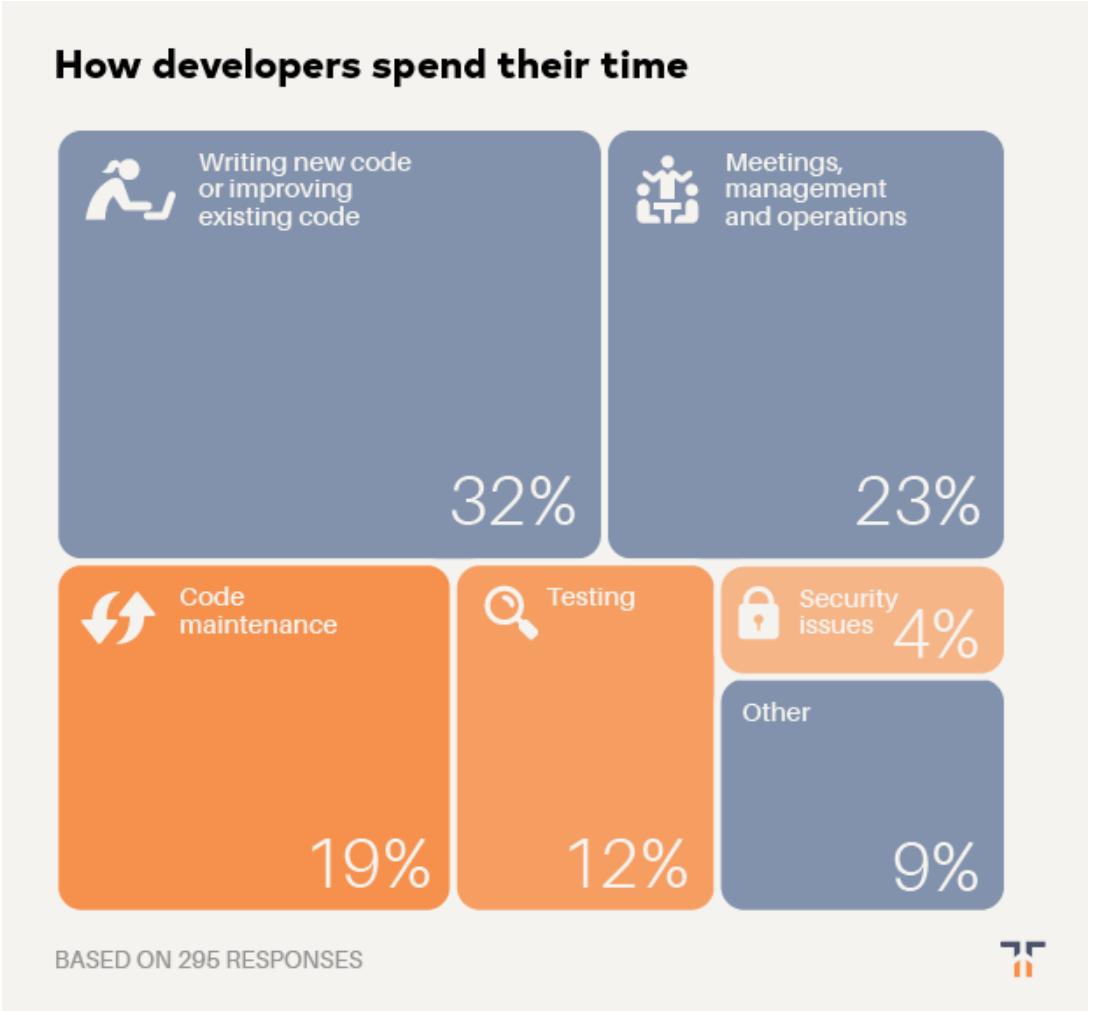
Tiempo de inactividad acumulado

(*) Analizó 606 fallas de software de 314 empresas
(*) Informe realizado por la empresa de pruebas de software Tricentis

https://www.tricentis.com/blog/1-1-trillion-in-assets-impacted-by-software-defects-a-software-testing-fail/

Los errores en el Software

Desde el tiempo de un desarrollador

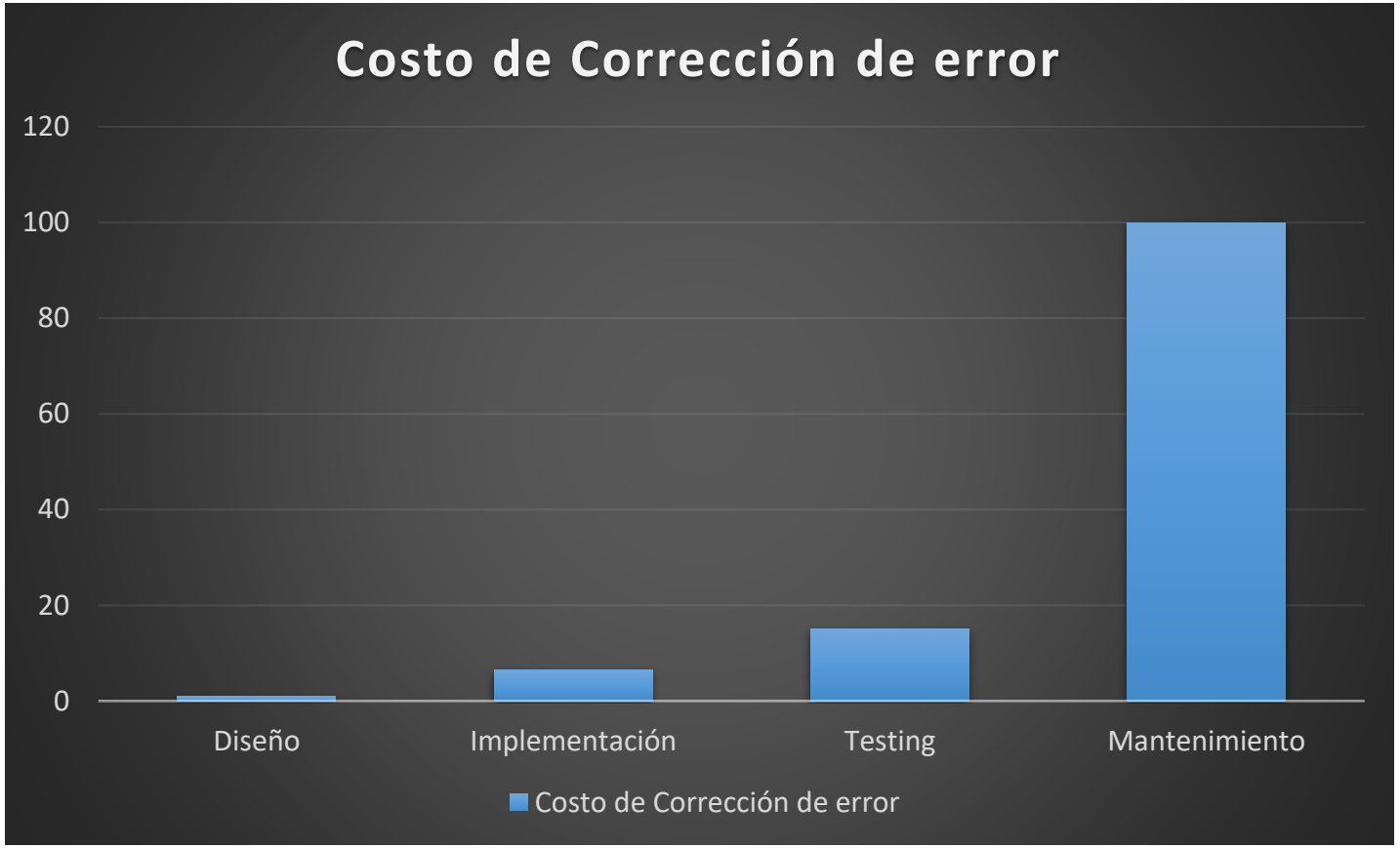


(*)junio de 2019, Tidelift y The New Stack

(*) https://thenewstack.io/how-much-time-do-developers-spend-actually-writing-code/

Los errores en el Software

Considerando el ciclo de vida de una aplicación



* IBM System Science Institute, costo relativo de corregir errores de software durante el ciclo de vida de una aplicación

Los errores en el Software

Todos se equivocan

Vuelo 501 de Ariane 5

4 de junio de 1996

$ 508 millones

Conversión de un tipo 64 bits en un espacio de 16 bits.

Manutención infantil

Reino Unido, 2004

$1.000 millones de dólares

Errores de migración de datos.

Mars Climate Orbiter (NASA)

Marte en 1998

$ 125 millones

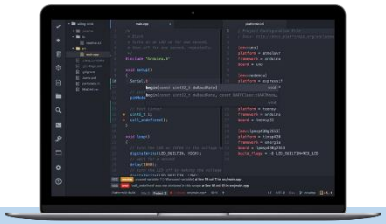
Error de conversión de unidades

Ciclo de vida de una Aplicación

Considerando el ciclo de vida de una aplicación desde el proyecto

Ciclo de vida de una Aplicación

Etapas destacables dentro del ciclo de vida de una aplicación



DESARROLLO

Tiempo de diseño

COMPILACIÓN

Tiempo de compilación

EJECUCIÓN

Tiempo de Ejecución

Alguno de los errores que se pueden producir se encuentran:

- Algoritmos mal usados o diseñados
- Errores de lógica

Alguno de los errores que se pueden producir se encuentran:

- Errores de sintaxis, como por ejemplo: Falta un “;” faltan símbolos de cierre.
- Falta de alguna librería

Alguno de los errores que se pueden producir se encuentran:

- División entre cero.
- Asignación forzadas de tipos.
- Acceso a memoria restringida

Tipos de errores

Errores de sintaxis → **Tiempo de compilación**
Al escribir el programa

Errores semánticos → **Tiempo de diseño**
Algoritmo incorrecto

Errores de ejecución → **Tiempo de Ejecución**
Errores que suceden con el programa funcionando

https://uniwebsidad.com/libros/algoritmos-python/capitulo-12

Excepciones

Los errores en tiempo de ejecución son llamados comúnmente *excepciones*

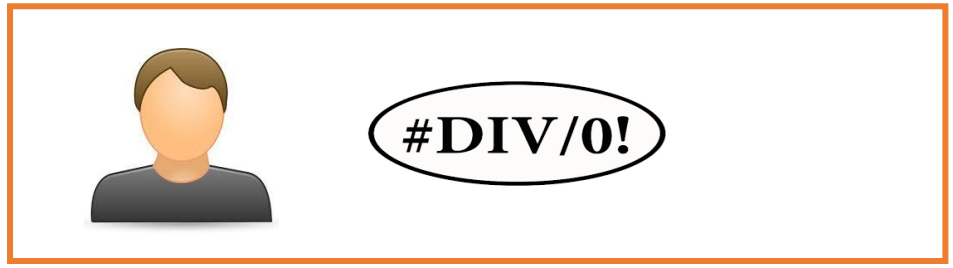
Excepciones

Situaciones que pueden provocar una Excepción

Falta de un recurso

Fallos de Hardware

Usos Indebidos



Excepciones

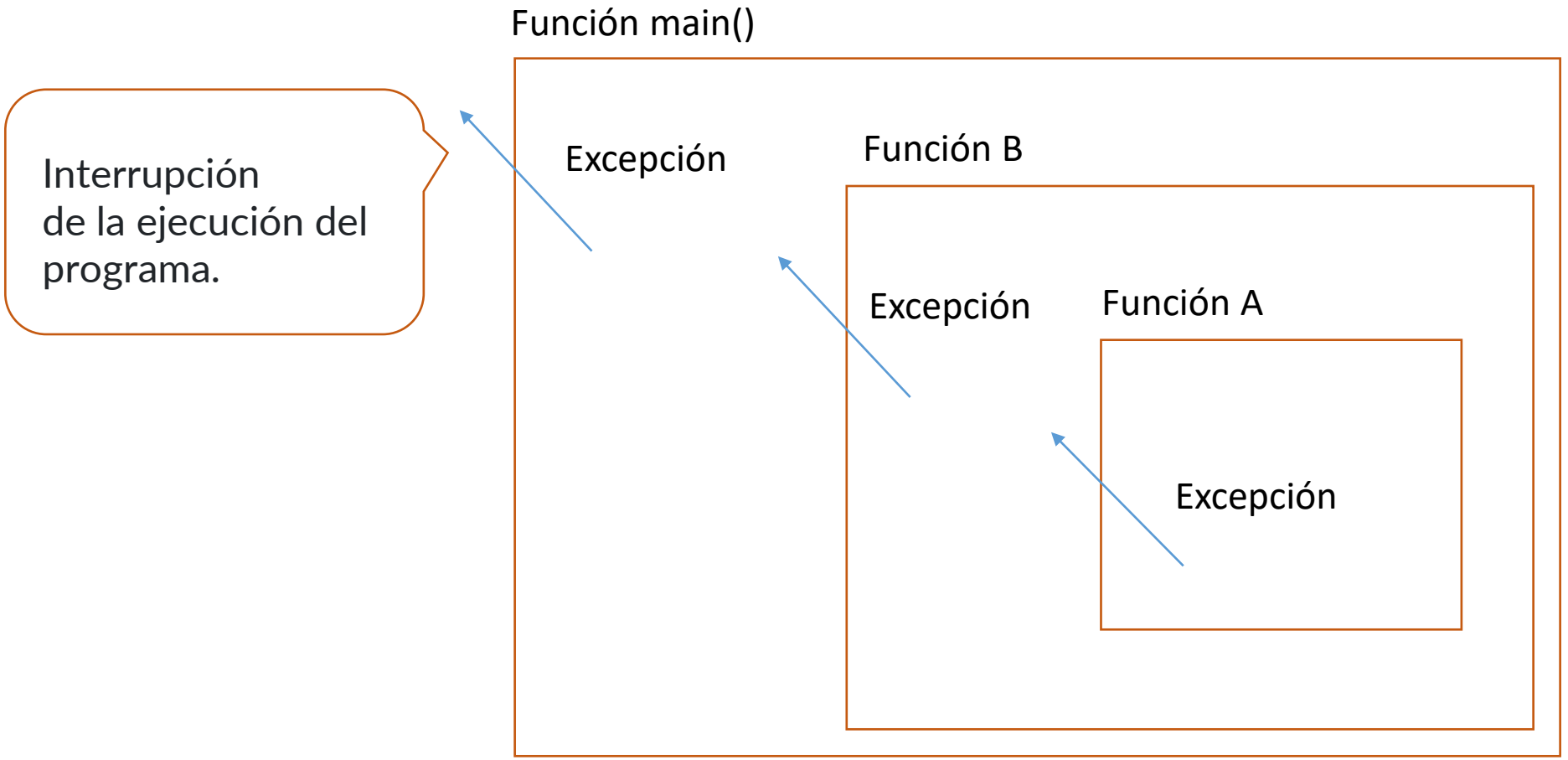
Tipo de excepciones

Implícitas → Son aquellas definidas por el lenguaje o el framework

Explícitas → Definidas por el programador.

Excepciones

Propagación de excepciones



Excepciones

Gestión de excepciones

Utilización un ciclo try-catch()

Try → Pone en alerta al programa sobre código que puede lanzar una excepción.

catch → Captura y manejar cada excepción que se lance

finally → Bloque de código opcional que se ejecutará haya o no excepciones.

Excepciones

Beneficios de la Gestión de excepciones

- Detectar errores y hacer una posible recuperación.
- Limpieza y salida elegante en caso de errores no esperados.
- Dejar registro de la ocurrencia de un determinado error sucedió en tiempo de ejecución.
- Propagación sistemática de errores en una cadena de llamadas dinámicas

Excepciones

Capturando una excepción

```
try
{
        // Código que se intenta ejecutar.
}
Catch
{
        // Código que se ejecuta en caso que se produzca una excepción.
}
```

Ejemplo 1

Excepciones

Capturando una excepción

```
try
{
        // Código que se intenta ejecutar.
}
Catch (Exception ex)
{
        // Código que se ejecuta en caso que se produzca una excepción.
}
```

Ejemplo 1.2

Excepciones

El bloque try separa el código que podría verse afectado por una excepción.
Los bloques catch asociados se usan para controlar las excepciones resultantes.
Los bloques finally se ejecutaran siempre al salir y de try o de un catch

```
try
{
        // Código que se intenta ejecutar
}
catch (Exception ex)
{
        // Código que se ejecuta en caso que se produzca una excepción.
}
finally
{
        // Codigo que se ejecuta siempre
}
```

Ejemplo 2

Excepciones

Lanzar excepciones

- EN C# La palabra reservada **Throw** nos permite lanzar una excepción

```
try
{
        // [...] código controlado
}
catch(Exception ex)
{
        throw;
}
```

Ejemplo 3

Excepciones

Lanzar excepciones específicas

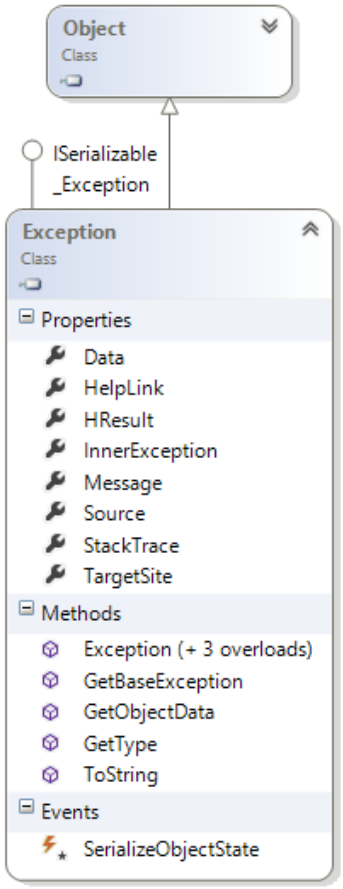
En diseño de código, es recomendable agregar información a una excepción que se vuelve a iniciar para proporcionar más información durante la depuración.

```
try
{
        // [...] código controlado
}
catch(Exception ex)
{
        throw new Exception("Contexto del error ", ex);
}
```

Ver. Ejemplo N3

La Clase Exception

System.Exception



Exception Es la clase base de todas las excepciones

Algunas Propiedades destacables:

Message – Mensaje más amigable para el usuario about error

Source – Nombre de la Fuente del error (application or object)

InnerException – Si es llamada por otra, tiene la lista de llamadas internas

StackTrace – Pila de llamadas del putno de la excepción

TargetSite – nombre que Lanza la Excepción

HelpLink – Dirección URL a información relacionada con la Excepción

Data – diccionario con información adicional sobre la excepción.

Excepciones

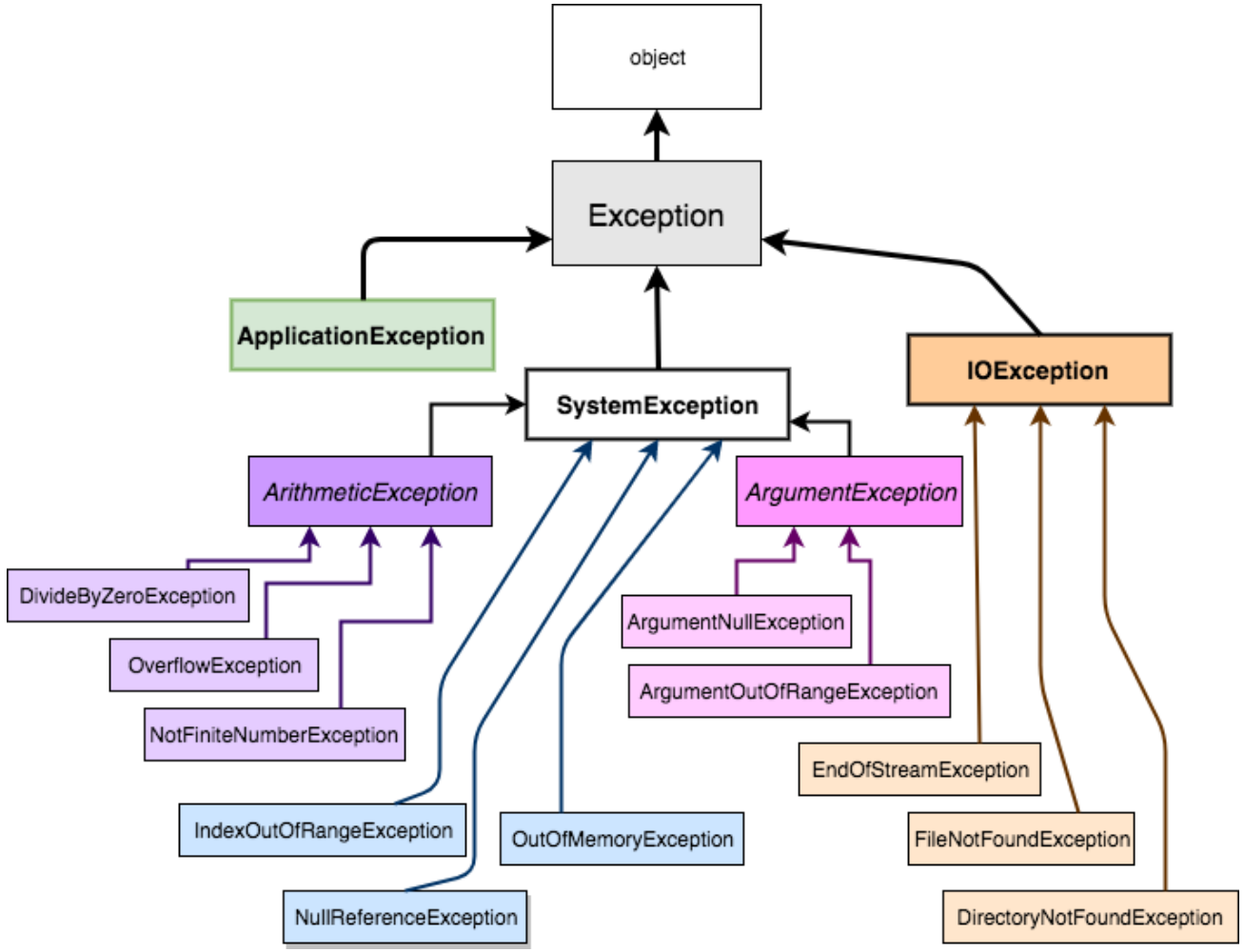
Información que podemos recuperar de la Clase Exception

```
try
{
         MetodoQueProvocaUnaExcepcion();
}
catch (Exception ex)
{

   var mensaje = "Error message: " + ex.Message; // Qué ha sucedido
   if (ex.InnerException != null) // Información sobre la excepción
   {
        mensaje = mensaje + " Inner exception: " + ex.InnerException.Message;
   }
   mensaje = mensaje + " Stack trace: " + ex.StackTrace;  // Dónde ha sucedido
}
```

https://docs.microsoft.com/es-es/dotnet/standard/exceptions/exception-class-and-properties

La Clase Exception

Excepciones derivadas



La Clase Exception

Filtros de excepción

```
string s = Console.ReadLine();

try
{
        Int32.Parse(s);
        Console.WriteLine(“ingresó un número int32 válido {0}.", s);
}
catch (FormatException e)
{
        Console.WriteLine(“ingresó un valor inválido {0}.", s);
}
catch (OverflowException e)
{
        Console.WriteLine($“Ingresó un número demasiado Grande  {e}");
}
```

Ver. Ejemplo N3

Excepciones

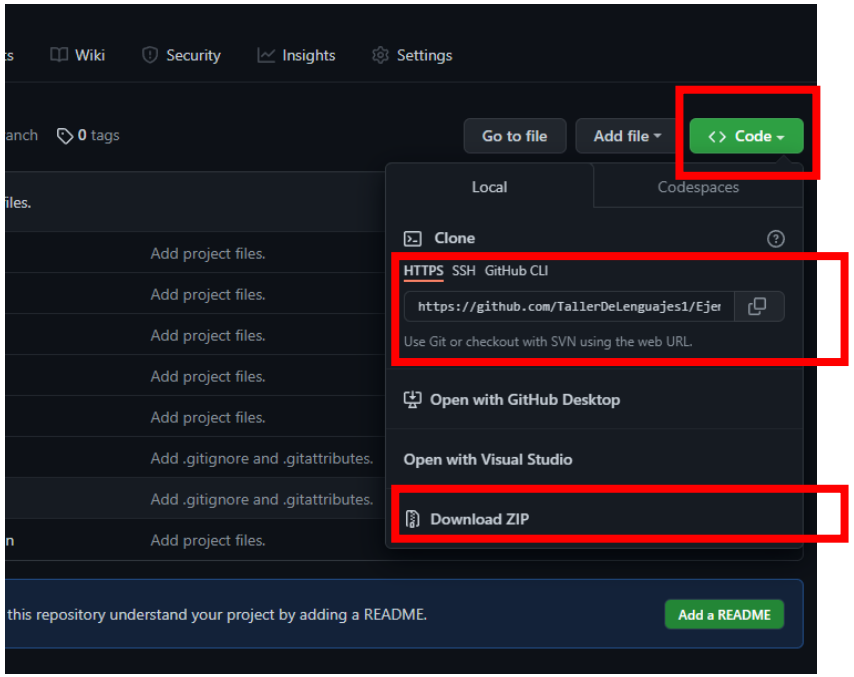
Clase Exception – Creando Una Excepción (Excepciones definidas por el usuario)

```
using System;

public class EmployeeListNotFoundException : Exception
{
   public EmployeeListNotFoundException()
   {
   }

   public EmployeeListNotFoundException(string message)
      : base(message)
   {
   }

   public EmployeeListNotFoundException(string message, Exception inner)
      : base(message, inner)
   {
   }
}
```

Ejemplos Utilizados

Ejemplos de código de la clase

https://github.com/TallerDeLenguajes1/EjemplosConcurso



Click en code

Para Clonar el repositorio, se puede copiar este link

Para Descargar el repositorio, se debe hacer click en **Download ZIP**

Propuesta Práctica

- Identificar en el siguiente código, el tipo de excepción y que lo provocó.
- Proponer una solución para los errores presentes.

https://classroom.github.com/a/fdq5qLA7

Bibliografía

https://uniwebsidad.com/libros/algoritmos-python/capitulo-12

https://www.researchgate.net/figure/IBM-System-Science-Institute-Relative-Cost-of-Fixing-Defects_fig1_255965523

Dawson, Maurice & Burrell, Darrell & Rahim, Emad & Brewster, Stephen. (2010). Integrating Software Assurance into the Software Development Life Cycle (SDLC). Journal of Information Systems Technology and Planning. 3. 49-53.

https://devops.com/survey-fixing-bugs-stealing-time-from-development/

https://www.businesswire.com/news/home/20210216005484/en/Rollbar-Research-Shows-That-Traditional-Error-Monitoring-Is-Missing-the-Mark

https://docs.microsoft.com/es-es/dotnet/standard/exceptions/

https://www.quora.com/How-long-does-it-take-you-to-fix-a-program-bug