Conceptos y Paradigmas de Lenguajes de Programación - Práctica 9

EJERCICIO 1

Excepción: es una condición inesperada o inusual que surge durante la ejecución del programa y no puede ser manejada en el contexto local. Condición anómala e inesperada que necesita ser controlada.

EJERCICIO 2

Para que un lenguaje trate excepciones debe proveer mínimamente:
- Un modo de definirlas
- Una forma de reconocerlas
- Una forma de lanzarlas y capturarlas
- Una forma de manejarlas especificando el código y respuestas
- Un criterio de continuación

No, no todos los lenguajes lo proveen, por ejemplo **C** estándar **no provee** manejo de excepciones.

EJERCICIO 3

Cuando un lenguaje de programación **no proporciona un mecanismo de manejo de excepciones**, puede ser más difícil y menos estructurado manejar errores y situaciones excepcionales en el código. Sin un manejo adecuado de excepciones, los errores pueden propagarse sin control, lo que puede llevar a un comportamiento inesperado y a una mayor dificultad para depurar el código.

Aunque no se disponga de un mecanismo de manejo de excepciones incorporado en el lenguaje, es posible simularlo utilizando técnicas alternativas. Una forma común de simular el manejo de excepciones es mediante el **uso de estructuras de control condicionales y funciones de retorno de errores.**

Es importante tener en cuenta que este enfoque simulado del manejo de excepciones puede resultar más propenso a errores y puede llevar a un código **más complicado y difícil de mantener** en comparación con un lenguaje que ofrece un manejo de excepciones nativo. Además, la simulación no proporcionará todas las funcionalidades avanzadas de los mecanismos de manejo de excepciones, como la capacidad de capturar excepciones específicas o realizar acciones de limpieza antes de propagar el error.

EJERCICIO 4 - 01

Reasunción: cuando se produce una excepción, se maneja y al terminar de ser manejada se devuelve el control a la sentencia siguiente de donde se levantó la excepción. Lenguajes: PL/1.

Terminación: cuando se produce una excepción, el bloque donde se levantó la excepción es terminado y se ejecuta el manejador asociado a la excepción. Lenguajes: ADA, CLU, C++, Java, Python o PHP.

EJERCICIO 4 - 02

Reasunción

	PL/I
¿Cómo se define?	Con instrucciones como **'ON condition DO; ... END;'**
¿Cómo se lanza?	Explícitamente **'SIGNAL condition'.** Automáticamente cuando ocurre una condición predefinida (por ej. división por cero activa ON ZERODIVIDE).
¿Cómo se maneja?	El bloque **'DO/BEGIN ... END'** se ejecuta al ocurrir la condición.
Criterio de continuación	**Reasunción**: Luego de ejecutarse el manejador, el control vuelve a la instrucción donde se generó la excepción.

Terminación

	ADA	C++	JAVA	PYTHON	PHP
¿Cómo se define?	**Nombre: exception;** Se define con bloques **'begin .. exception .. end'.**	Se define con bloques **try-catch.**	Se utiliza **try-catch-finally.**	Con bloques **try-except-finally.**	Con bloques **try-catch-finally.**
¿Cómo se lanza?	Con **'raise NombreDeLa Excepción;'.**	Con la instrucción **'throw'.**	Con **'throw new Exception();'.**	Con **'raise Exception'.**	Con **'throw new Exception();'.**
¿Cómo se maneja?	Dentro del bloque '**exception**' usando '**when**'.	Con bloques '**catch**' que capturan excepciones según su tipo.	Con bloques '**catch**' específicos para cada tipo de excepción.	Con bloques '**except**' que capturan excepciones por tipo.	Con bloques '**catch**'.
Criterio de continuación	**Terminación**: El bloque se termina y el control pasa a después del bloque que lanzó la excepción.	**Terminación**: El control salta al bloque catch y luego continúa después del try-catch.	**Terminación**: El flujo pasa al bloque catch y luego continúa después del bloque try-catch-finally.	**Terminación**: El control pasa al bloque except y luego continúa después del try.	**Terminación**: El flujo salta al catch y luego continúa normalmente.

EJERCICIO 4 - 03

El modelo por **reasunción** es mas inseguro dado que cuando surge una excepción no termina, sigue ejecutando donde se quedo y puede que eso acarree errores en la ejecución.

EJERCICIO 5-14