EXCEPCIONES EN JAVA

* Un lenguaje O.O. debe proporcionar mecanismos para gestionar excepciones.
* Como mínimo al gestionar una excepción debe resolver:
  + Informar al usuario lo ocurrido.
  + Guardar el trabajo del usuario.
  + Permitirle salir de forma amigable.
* Los errores pueden ocurrir por:
  + Malas entradas de información o datos por parte del usuario.
  + Errores de hardware.
  + Limitaciones físicas del hardware.
  + Errores de lógica en la codificación.
* Jerarquía de excepciones.

X Programador

√ U.M.

Object

Throwable

Error

Exception

√ Programador

-InternalError

-OutOfMemoryError

-StackOverFlowError

-UnknownError

-LinkageError

-ThreadDeath

-AWTError

-VirtualMachineError

-AbstractMethodError

-IllegalAccessError

-InstantiationError

-NoSuchFieldError

-NoSuchMethodError

-ClassCircularityError

-ClassFormatError

-IncompatibleChangeError

-NoClassDefFoundError

-UnsatisfiedLinkError

-VerifyError

Error

Exception

-ClassNotFoundException

-ClassNotSupportedException

-IllegalAccessException

-InstantiationException

-InterruptedException

-RuntimeException

-NoSuchMethodException

-AWTException

-SQLException

-IOException

-EmptyStackException

-NoSuchElementException

-IllegalArgumentException

-IndexOutOfBoundsException

-ArithmeticException

-ArrayStoreException

-ClassCastException

-IllegalMonitorStateException

-NegativeArraySizeException

-NullPointerException

-SecurityException

-IllegalThreadStateException

-NumberFormatException

-ArrayIndexOutOfBoundsException

-StringIndexOutOfBoundsException

-EOFException

-FileNotFoundException

-InterruptedIOException

-UTFDataFormatException

-MalFormatException

-ProtocolException

-SocketException

-UnknownHostException

-UnknownServiceException

* Atrapar excepciones

1 bloque try{}

1 o más bloques catch{}

1 bloque finally{}

public int[] crearArreglo(int n) throws NegativeArraySizeException { try{

if(n < 0)throws new DimensionNegativaDeArregloException();

return new int[n]; int n = 10;

} int[] arreglo = crearArreglo(n);

Public int dividir(int a,intb){ int c = dividir(n,arreglo[2]);

return a/b; }catch(DimensionNegativaDeArregloException dndae){

} System.out.println(dndae)

flujo

dndae.printStackTrace();

}catch(ArithmeticException ae){

ae.printStackTrace();

}catch(ArrayIndexOutOfBoundsException aioobe){

Aioobe.printStackTrace();

}catch(Exception e){

e.printStackTrace();

}finally{

Se puede usar para cerrar flujos, cerrar archivos y . liberar flujos}

* Lanzar excepciones

throws() = muchas && throw() = una

public class DimensionNegativaDeArregloException extends NegativeArraySizeException{

private String mensaje;

public DimensionNegativaDeArregloException{

this(“n debe ser mayor a 1”);

}

public DimensionNegativaDeArregloException(String mje){

super();

mensaje = mje;

}

public String toString(){

return mensaje;

}

}