Manejo de Excepciones

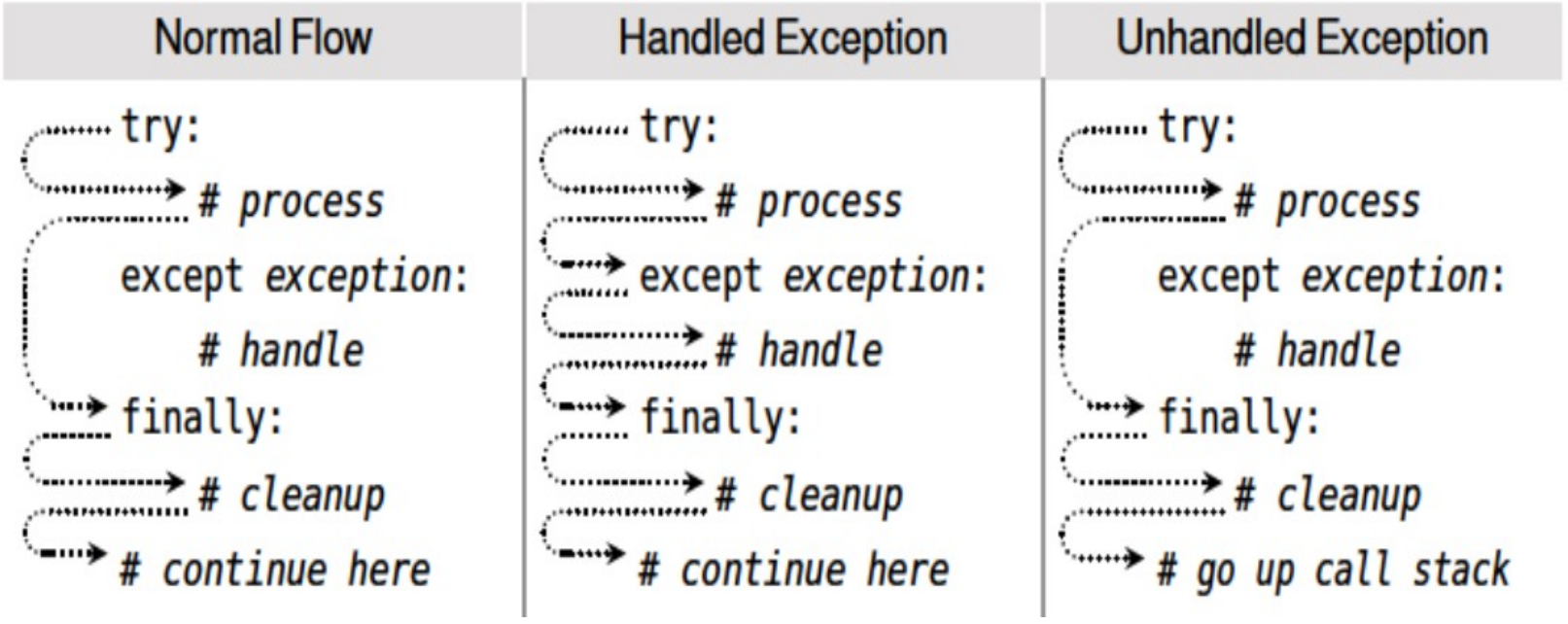
- Hay dos tipos de errores durante la ejecución de un programa:
  - Errores de sintaxis:
    - Los errores de sintaxis ocurren cuando se escribe el código incorrectamente.
    - En tales casos, la línea errónea es repetida por el analizador con una flecha apuntando a la primera ubicación en donde el error fue detectado.
  - Excepciones:
    - Estos ocurren durante la ejecución de un programa cuando algo inesperado sucede. Por ejemplo, división por cero.
    - Cuando no estás manejando excepciones apropiadamente, el programa se cerrará de manera abrupta ya que no sabe que hacer en tales casos.

Manejo de Excepciones

```
try:
    <sentencias>
except excepción1 [as variable1]:
    <sentencias>
...
except excepciónN [as variableN]:
    <sentencias>
except (exA, exB, ...)  [as variable]:
    <sentencias>
except:
    <sentencias>
else:
    <sentencias>
finally:
    <sentencias>
```

- **Try**: permite controlar las excepciones dentro de un bloque de código.
- **Except**: permite ejecutar código si ocurrió alguna excepción.
- **Else**: permite ejecutar código si no ocurrieron excepciones.
- **Finally**: permite ejecutar código independientemente de si ocurrieron o no excepciones.

Manejo de Excepciones



Manejo de Excepciones

```
while True:
    try:
        x = int(input("Ingrese un número: "))
    except ValueError:
        print("El valor ingresad no es un entero.")
    else:
        print("Calculando 50 /", x,"Resultado:", 50/x)
    finally:
        print("Ya hice todo lo necesario.")
```

Manejo de Excepciones

- En ciertas ocasiones es deseable generar una excepción:
  - Si estamos dentro de un bloque except, se puede lanzar una excepción sin especificar que excepción.

    raise
  - Especificando un excepción (estándar o personalizada) y un argumento:

    raise <TipoDeError>(<mensaje>)

```
x = -1
if x < 0:
    raise Exception("x tiene un valor negativo")
```

Manejo de Excepciones

- Las excepciones personalizadas son clases que heredan de otra excepción:

```
class exceptionName(baseException):
    <sentencias>
```

- Ejemplo:

```
class MiException(Exception):
    def __init__(self, mensaje):
        super().__init__(mensaje)

raise MiException("Usando una excepción personalizada")
```