12.2.1. Manejo de excepciones

Para el manejo de excepciones los lenguajes proveen ciertas palabras reservadas, que nos permiten manejar las excepciones que puedan surgir y tomar acciones de recuperación para evitar la interrupción del programa o, al menos, para realizar algunas acciones adicionales antes de interrumpir el programa.

En el caso de Python, el manejo de excepciones se hace mediante los bloques que utilizan las sentencias `try`, `except` y `finally`.

Dentro del bloque `try` se ubica todo el código que pueda llegar a *levantar* una excepción, se utiliza el término *levantar* para referirse a la acción de generar una excepción.

A continuación se ubica el bloque `except`, que se encarga de capturar la excepción y nos da la oportunidad de procesarla mostrando por ejemplo un mensaje adecuado al usuario. Veamos qué sucede si se quiere realizar una división por cero:

```
>>> dividendo = 5 >>> divisor = 0 >>> dividendo / divisor Traceb
ack (most recent call last):   File "<stdin>", line 1, in <modul
e> ZeroDivisionError: integer division or modulo by zero
```

En este caso, se levantó la excepción `ZeroDivisionError` cuando se quiso hacer la división. Para evitar que se levante la excepción y se detenga la ejecución del programa, se utiliza el bloque `try-except`.

```
>>> try: ...     cociente = dividendo / divisor ... except: ...
    print "No se permite la división por cero" ...
```

No se permite la división por cero

Dado que dentro de un mismo bloque `try` pueden producirse excepciones de distinto tipo, es posible utilizar varios bloques `except`, cada uno para capturar un tipo distinto de excepción.

Esto se hace especificando a continuación de la sentencia `except` el nombre de la excepción que se pretende capturar. Un mismo bloque `except` puede atrapar varios tipos de excepciones, lo cual se hace especificando los nombres de la excepciones separados por comas a continuación de la palabra`except`. Es importante destacar que si bien luego de un bloque `try` puede haber varios bloques `except`, se ejecutará, a lo sumo, uno de ellos.

```
try:     # aquí ponemos el código que puede lanzar excepciones e
xcept IOError:     # entrará aquí en caso que se haya producido
    # una excepción IOError except ZeroDivisionError:     # entr
ará aquí en caso que se haya producido     # una excepción ZeroD
ivisionError except:     # entrará aquí en caso que se haya prod
ucido     # una excepción que no corresponda a ninguno     # de
los tipos especificados en los except previos
```