PYTHON

Fundamentos de Python 2

Cadenas, Métodos de Listas y Excepciones

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

CADENAS

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- ASCII (*American Standard Code for Information Interchange*)
- 8 bits - 256 caracteres distintos.
  - 128 primeros – alfabeto latino.
  - 128 restantes – en función de la página de códigos. Permiten representar caracteres de otros idiomas pero hay problemas de ambigüedad.
- Concepto: punto de código. Es el número que compone un carácter.
- Entre una letra minúscula y mayúsculas existen 32 números de diferencia, siendo 32 el número del espacio.

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- i18n→Internacionalización.
- UNICODE. Alternativa a las páginas de código.
  - Caracteres únicos asignados a más de 1.000.000 de puntos de código.
    - 128 primeros igual que ASCII.
    - 256 primeros igual que la página de códigos ISO/IEC 8859-1.
  - UCS-4 (Universal Character Set) es una de las implementaciones de UNICODE.
    - Utiliza 32 bits (siempre) → desperdicio de espacio.
    - Cada código es un punto de código UNICODE.

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- UTF-8 (Unicode Transformation Format). Implementación de UNICODE.
  - Optimiza el espacio. Utiliza los bits necesarios para cada punto de código.
  - Puede utilizar BOM → Marca no imprimible que determina el tipo de código que maneja un archivo.
- Python3 es compatible con UNICODE y UTF-8:
  - Se puede utilizar para nombrar elementos de código.
  - Se puede utilizar en las entradas y salidas de los programas.
  - Python está INTERNACIONALIZADO.

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- Cadenas de caracteres:
  - Son secuencias **inmutables**.
  - En una única línea, delimitadas por ' o "
  - En varias líneas, delimitadas por ' ' ' o """
  - La función **len** cuenta espacios, saltos de línea y tabulaciones.
  - Sobrecarga de operadores aplicados en cadenas:
    - + → Concatenación (atención a los tipos)
    - * → Replicación (el orden de los operadores no importa)
    - Atajos admitidos:
      - +=
      - *=

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- Cadenas de caracteres:
  - Función ord(*caracter*) → Proporciona el punto de código ASCII/UNICODE de un carácter.
    - Si se equivoca el tipo de parámetro → TypeError.
  - Función chr*(entero)* → Proporciona el carácter de un punto de código.
  - Las cadenas admiten indexación → Acceder a sus componentes por posición, utilizar *slicing*, recorrerlas con un *for*.
  - Admiten el uso de los operadores **in** y **not in**.

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- Cadenas de caracteres:
  - Al ser inmutable:
    - No admite el uso de **del** aplicado a una parte.
    - No admite **append**.
    - No admite **insert.**
  - Se puede utilizar con las funciones **min**() y **max**() → No puede estar vacía (ValueError).
  - Método **index**()→Busca una subcadena y devuelve la posición de la primera aparición o ValueError si no existe.
  - Función **list**() convierte la cadena en una lista con los caracteres.
  - Función **count**() proporciona el número de elementos de la cadena.

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- Cadenas de caracteres:
  - Al ser inmutable:
    - No admite el uso de **del** aplicado a una parte.
    - No admite **append**.
    - No admite **insert.**
  - Se puede utilizar con las funciones **min**() y **max**() → No puede estar vacía (ValueError).
  - Función **list**() convierte la cadena en una lista con los caracteres.
  - Función **count**() proporciona el número de veces que aparece una subcadena en una cadena.

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- Cadenas de caracteres:
  - Métodos:
    - Método **index**()→Busca una subcadena y devuelve la posición de la primera aparición o ValueError si no existe. Permite delimitar el ámbito de la búsqueda.
    - Método **find**()→Igual que **index**(), pero si no existe devuelve **-1**. Para determinar si existe o no un elemento, utilizar operador **in** (es más rápido). Permite delimitar el ámbito de la búsqueda.
    - Método **capitalize**()→Convierte el primer carácter a mayúscula (si es alfabético) y el resto a minúscula.
    - Método **center***(n)*→Ocupando *n* espacios, centra la cadena.
    - Métodos **endswith**(*subcadena*), **startswith**(*subcadena)*→Indica si la cadena termina o empieza en la subcadena.

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- Cadenas de caracteres:
  - Métodos:
    - Método **isalnum**()→Determina si todos los caracteres de una cadena son dígitos y caracteres alfabéticos.
    - Método **isalpha**()→Determina si todos los caracteres de una cadena son caracteres alfabéticos.
    - Método **isidigit**()→Determina si todos los caracteres de una cadena son dígitos.
    - Método **islower**()→Determina si todos los caracteres de una cadena son caracteres alfabéticos escritos en minúsculas.
    - Método **isupper**()→Determina si todos los caracteres de una cadena son caracteres alfabéticos escritos en mayúsculas.
    - Método **isspace**()→Determina si todos los caracteres de una cadena son espacios en blanco.

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- Cadenas de caracteres:
  - Métodos:
    - Método **join**()→Construye una cadena a partir de los elementos de una lista (o tupla, o conjunto –en este caso sin orden-)
      - *cadena_base*.**join**(*secuencia*) → Genera una cadena con los elementos de *secuencia* separados por el contenido de *cadena_base.*
      - Si algún elemento de la secuencia no es cadena→TypeError.
    - Métodos **lower**() y **upper**()→Generan una copia de la cadena convertida a minúscula o mayúscula.
    - Métodos **lstrip**(),**rstring**(), **strip**()→Eliminan los espacios en blanco en el principio de la cadena (*left*), en la derecha *(right*) o en ambos extremos.

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- Cadenas de caracteres:
  - Métodos:
    - Método **replace**(*subcadena1, subcadena2, número_reemplazos)*→Reemplaza todas las apariciones de *subcadena1* por *subcadena2*. Si se indica, limita los cambios al número de reemplazos indicado.
    - Método **rfind**()→Igual que **find**(), pero comenzando desde el final de la cadena.
    - Método **split**()→Divide una cadena en una lista de subcadenas. Utiliza el espacio como separador, pero se puede indicar.
    - Método **swapcase**()→Convierte mayúsculas a minúsculas y viceversa.
    - Método **title**()→Convierte la primera letra de cada palabra a mayúscula y el resto a minúscula.

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- Cadenas de caracteres:
  - Comparación:
    - Operadores ==, !=, >, >=, <, <=
    - Compara valores de puntos de código. Letras mayúsculas menores que las minúsculas.
    - Se compara el primer carácter diferente.
    - Cadena2 con los mismos caracteres más uno que Cadena1 → Cadena2 es mayor (longitud).
    - Comparar cadenas con números:
      - Posible con == y != (siempre False y True)
      - TypeError con el resto
    - Conceptos: distancia **Hamming**, distancia **Levenshtein** y algoritmo **Soundex**.

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

MÉTODOS DE LISTAS

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- Cadenas de caracteres:
  - Ordenar listas que contienen cadenas:
    - Función sorted() → Genera una nueva lista ordenada.
    - Método sort() → Ordena la lista.
  - Conversiones de tipo:
    - Funciones str(), int() y float(). ValueError si no es posible.

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

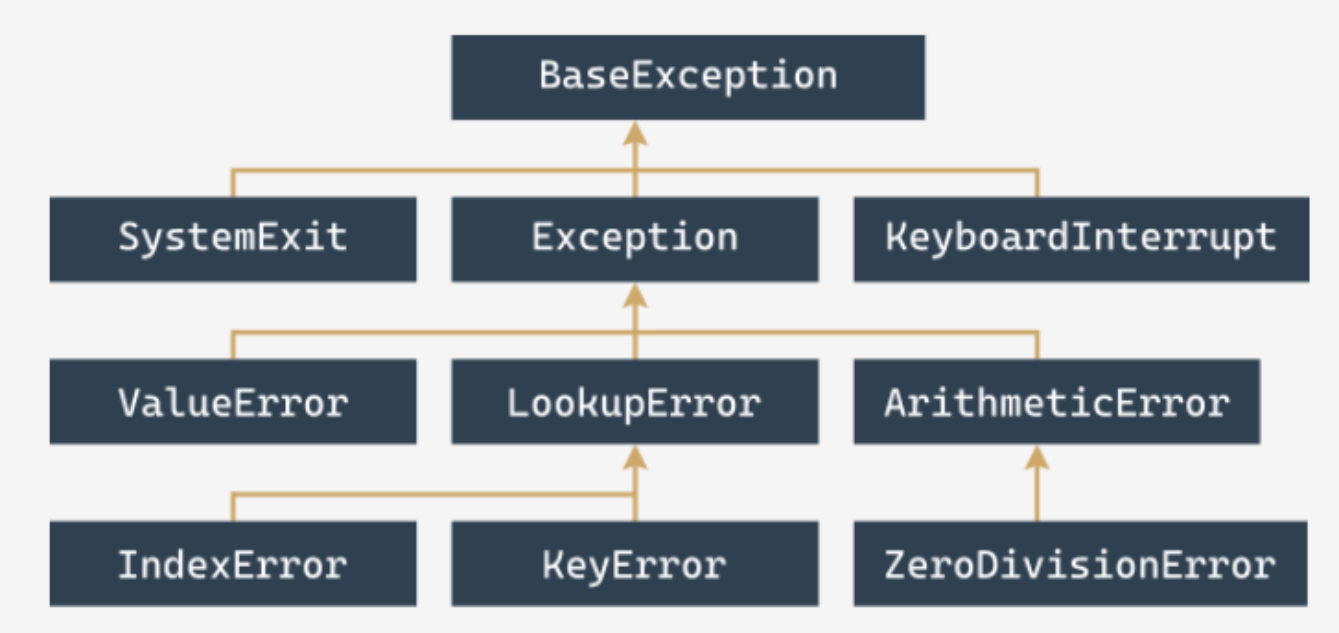
EXCEPCIONES

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- Excepciones:
  - Python incluye 63 excepciones integradas.
  - Bloque try-except
    - Except solo (captura todas excepciones)
    - Except con excepción
    - Except con excepción y objeto
  - La instrucción raise
    - Aplicada a la clase: "raise ZeroDivisionError"
    - Aplicada a una instancia: "raise ZeroDivisonError(*descripción*)"

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- Excepciones:
  - Jerarquía de excepciones. La importancia del orden en la captura.
  - Las excepciones finales se les denomina **concretas.**



Fuente del diagrama: edube.org

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- Aserciones:
  - La instrucción **assert** y la excepción **AssertionError**
  - assert *expresión* genera una **AssertionError** si el resultado es True, un valor numérico distinto de 0, o un valor distinto de None.
  - Es un complemento a la validación de datos.

CADENAS, MÉTODOS DE LISTAS Y EXCEPCIONES

- Algunas excepciones:
  - ArithmeticError(Exception(BaseException)).
  - AssertionError(Exception(BaseException)).
  - BaseException().
  - IndexError(LookupError(Exception(BaseException)).
  - KeyboardInterrupt(BaseException).
  - MemoryError(Exception(BaseException)).
  - OverflowError(ArithmeticError(Exception(BaseException))).
  - ImportError(StandardError(Exception(BaseException))).
  - KeyError(LookupError(Exception(BaseException))).