Desarrollo de herramientas de software: Introducción a Python

16 de septiembre de 2021

1. Python como calculadora

1. Dado dos números enteros imprimir por pantalla el producto de los mismos. Si el producto es más grande que 10000, imprimir su suma
2. Realizar un programa que calcule el desglose mínimo en billetes y monedas de una cantidad exacta de pesos. Hay billetes de $1000, $500, $200, $100 y $500 y monedas de $10, $5 y $2 y $1. El desglose se realiza del más grande al más pequeño. . El desglose se realiza del más grande al más pequeño.
3. Diseñar un programa que, dados cinco puntos en el plano, determine cuál de los cuatro últimos es más cercano al primero.
4. Dado un rango de 10 números enteros, iterar desde el primero al último y en cada iteración imprimir:
   - Valor actual
   - Valor anterior
   - Valor siguiente
   - Suma acumulada

   Nota: se debe usar range() para generar el rango de números
5. Realizar un programa que permita determinar las raíces de un polinomio de segundo grado. Se deben contemplar los tres casos posibles

2. Strings

1. Considerando que se tiene un string cargado, mostrar por pantalla sólo las letras que se encuentren ubicadas en las posiciones pares
2. Solicitar el ingreso de un string por teclado junto a un número entero. Se deben eliminar los caracteres comprendidos el inicio del string y el número ingresado
3. Modificar el programa para que reciba dos números enteros que permitan definir el comienzo y el fin de caracteres a extraer del string original

4. Solicitar el ingreso de un número por teclado e imprimirlo por pantalla invertida. Ejemplo: in: 123 - out: 321
5. Recibir por teclado un string y contar la cantidad de mayúsculas, minúsculas y símbolos que se encuentran en la misma
6. Recibir por teclado una frase de más de 80 caracteres. Luego, recibir también una segunda palabra. Finalmente, imprimir cuantas veces aparece la segunda palabra en la frase del comienzo
7. Recibir por teclado una frase de más de 80 caracteres. Luego, recibir también una segunda palabra. Finalmente, imprimir cuantas veces aparece la segunda palabra en la frase del comienzo
8. Partiendo de un string, recibir un número entero y contar cuántas palabras son mas largas, mas cortas o iguales que el número ingresado
9. Diseñar un programa que recibiendo el nombre de un archivo, permita separar el nombre de su extensión
10. Diseñar un programa que lea dos cadenas de texto e imprima la mayor coincidencia entre ambas
11. Detectar si una palabra ingresada es palindromo o no
12. Detectar si una palabra es alfabética
13. Detectar si una palabra es alfabética
14. Luego de recibir por teclado un string que contenga letras y números, imprimir la cada número contenido en el string y la suma de los mismos
15. Diseñar un programa que permita validar que la estructura de una dirección de mail es correcta (debe tener un @ y al menos un punto). Por otro lado, realizar un validador de contraseñas (debe tener al menos una letra mayúscula, una minúscula, un número o un símbolo)

3. Listas

1. Considerando dos listas de números enteros:

   De cada una de las listas imprimir los números divisibles por 5

   - Calcular e imprimir la suma vectorial de ambas
   - Calcular e imprimir el producto punto
   - De cada una de las listas imprimir los números divisibles por 5
2. Crear una lista con 100 números aleatorios, luego se debe generar dos listas más: una que contenga los números pares de la primera y otra los impares
3. Ingresar un string por teclado y hacer dos listas: una que contenga las letras mayúsculas de la palabra y otra las minúsculas
4. Diseñar un programa que dada una lista con números positivos y negativos, sustituya los negativos por cero

5. Diseñar un programa permita la carga de dos matrices distintas de 3x3. Luego, se debe imprimir la suma y la multiplicación entre ambas

6. Diseñar un programa que tomando una lista de números enteros, genere una nueva eliminando los números repetidos de la primera

7. Crear una lista que contenga en su interior 27 listas, asociadas cada una a una letra del alfabeto español.
   Luego tomar por teclado 32 palabras y añadirlas cada una a la lista correspondiente a su primera letra.

4. Diccionarios

1. Convertir las siguientes listas a un diccionario:
   keys = ['Diez', 'Veinte', 'Treinta']
   values = [10, 20, 30]

2. Agregar la key çuarentaz su respectivo valor al ejercicio del apartado anterior

3. Borrar la key "veintez su respectivo valor al ejercicio del apartado anterior

4. Crear un diccionario llamado estudiante y luego la siguiente estructura:

   - Nombre: string
   - Apellido: string
   - Legajo: string
   - Materias que cursa: lista de strings

   Cargar valores e imprimirlo por pantalla

5. A partir del diccionario del ejercicio anterior, modificar el número de legajo y agregar dos materias nuevas en la lista de materias

6. Modificar el ejercicio anterior para que la lista de materias de cursa ahora sea un diccionario. Es decir, cada materia tiene asociado un código

7. A partir del diccionario del apartado anterior, generar una lista con 10 diccionarios. Cada uno de ellos representa un alumno distinto

5. Funciones

1. Crear una función en Python que imprima el valor y el ID de un objeto recibido en la misma. Luego crear un objeto lista, un entero y una tupla. Finalmente se deben imprimir estos tres objetos utilizando la función y sin hacerlo.
   Extraer conclusiones sobre como las funciones en Python reciben parámetros.

2. Crear una función que imprima las primeras n filas de un triángulo de Pascal.
   Referencias:

   - Construcción de un triángulo de Pascal.
   - Aplicación del triángulo de Pascal.

3. Escribir una función que reciba un string y luego calcule y retorne la cantidad de letras mayúsculas y minúsculas presentes en la misma

4. Escribir una función que reciba una lista y retorne una nueva lista con los elementos de la primera sin repeticiones

5. Escribir una función que acepte parámetros posicionales y kwargs. Luego la función debe imprimirlos indicando cómo fue recibido cada uno de los parámetros

6. Luego de instalar packete externo colour utilizando Anaconda (link a documentación), crear una función que permita imprimir un mensaje por consola. Se debe decorar la misma para permitir su utilización mediante tres formas:

   - Impresión de warnings: letras escritas en amarillo
   - Impresión de mensajes críticos: letras escritas en rojo
   - Impresión de normal: letras escritas en blanco

7. Luego de instalar packete externo time utilizando Anaconda (link a documentación), crear un decorador que permita medir el tiempo de ejecución de una función.
   Luego, se debe implementar una función que permita multiplicar elemento a elemento los siguientes objetos:

   - Dos listas del 100.000 elementos enteros
   - Dos tuplas del 100.000 elementos enteros
   - Dos listas del 100.000 elementos flotantes
   - Dos tuplas del 100.000 elementos flotantes

8. Escribir un programa que partiendo de una lista de números enteros, los clasifique en dos listas distintas: una con los números pares y otra con los impares. Resolver utilizando lambdas

9. Crear una lista de diez diccionarios donde cada uno de ellos contenga la información de un developer

- Nombre
- Edad
- Lenguajes manejados
- Años de experiencia

Se debe imprimir la lista de todas las formas mencionadas a continuación. Cada una de ellas debe ser implementada con una función lambda distinta.

- Alfabéticamente por nombre
- Edad
- Años de experiencia en forma ascendente
- Años de experiencia en forma descendente

10. Diseñar un programa que sume tres listas. Utilizar map() y lambdas.
11. Diseñar un programa calcule la raíz cuadrada de cada elemento de una lista. Utilizar map().
12. Diseñar un programa calcule la raíz cuadrada de cada elemento de una lista. Utilizar map().
13. Diseñar un programa que a partir de una lista genere otra con sólo los números positivos. Utilizar filter()
14. Diseñar un programa que a partir de un string, genere una lista con

6. Desafío Pythonico: el conjunto de Mandelbort

6.1. Desde Wikipedia

El conjunto de Mandelbrot es el más estudiado de los fractales. Se conoce así en honor al matemático Benoît Mandelbrot (1924-2010), que investigó sobre él en los años setenta.
Este conjunto se define por los números complejos c para los cuales la función representada en la ecuación 1 no diverge cuando es iterada desde $Z = 0$.

$$f_c(z) = Z^2 + C \tag{1}$$

Dicho de otra forma, si la sucesión recursiva representada por 2 es acotada, se dice que el número c corresponde al conjunto de Mandelbrot.

$$Z_{n+1} = Z_n^2 + C \tag{2}$$

Por ejemplo, si $c = 1$ se obtiene la sucesión $0, 1, 2, 5, 26...$ que claramente diverge. Es decir la suma representada por la ecuación 2 es divergente y 1 no pertenece al conjunto de Mandelbrot.
En cambio si $c = -1$, se obtiene la suceción $0, -1, 0, -1...$ que si es acotada y, por lo tanto, -1 pertenece al conjunto de Mandelbrot.

La figura 1 es una representación matemática del conjunto de Mandelbrot como subconjunto del plano complejo. Los puntos del conjunto se muestran en negro.

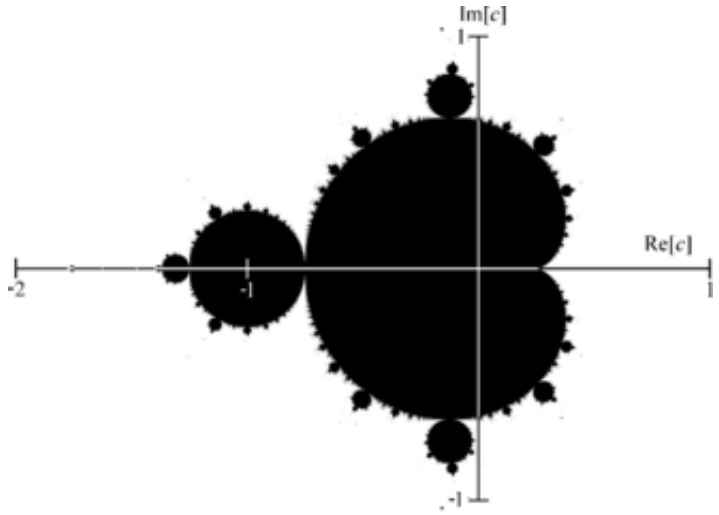

Figura 1: Diagrama de flujo estructura repetitiva while.

6.2. Implementación en Python

6.2.1. Módulos necesarios

Para poder generar una figura desde Python, se necesita instalar el módulo Pillow (link a documentación). Dentro del módulo Pillow, tendremos las herramientas necesarias para crear una imagen de un tamaño deseado en pixeles, como así también la capacidad para dibujar colorear un punto negro en una posición específica.

6.2.2. Pseudocódigo

A continuación se propone un pseudocódigo:

1. Definir una imagen de 800px por 600px (por ejemplo)
2. Generar dos estructuras repetitivas, una por cada uno de los ejes. Para este ejemplo, uno de ellos contará de 0 a 800 y el otro de 0 a 600
3. Convertir cada pixel a un número complejo
4. Calcular la serie de Mandelbrot para el número complejo generado. Este cálculo puede hacerse en una función y la misma debe tener dos condiciones de salida:
   - Llegar a un número máximo de iteraciones: por ejemplo 500
   - El módulo del número complejo generado es menor a 2 (la serie converge).
5. Si la función anterior sale porque se alcanzó el número de iteraciones, la serie no converge y se debe dibujar un pixel blanco. Caso contrario, la serie converge, es decir, las coordenadas representadas por los pixeles X;Y pertenecen al conjunto de Mandelbrot y se debe dibujar un pixel negro.

7. Expresiones regulares

1. Escribir un programa en Python que verifique si una cadena de texto comienza con una palabra
2. Escribir un programa en Python que a partir de un string, separe en una lista todas las palabras que contengan una .^{a}z finalicen en "b".
3. Escribir un programa en Python que a partir de un string, verifique si todas las lineas del mismo contienen un ; al final.
4. Escribir un programa en Python que a partir de un string, verifique cuales lineas del mismo contienen un signo +,-,*,/ y/o = en su interior.
5. Escribir un programa en Python que a partir de un string, extraiga las fechas con formato dd/mm/yyyy
6. Escribir un programa en Python que a partir de un string, extraiga todas las direcciones de mail en el mismo

8. Archivos

1. Listar todos los modos de apertura de un archivo y extraer conclusiones de cada uno
2. Listar todos los modos de apertura de un archivo y extraer conclusiones de cada uno
3. Escribir un programa en Python que lea todas las lineas de un archivo de texto
4. Escribir un programa en Python que lea un archivo linea a linea y almacene dicho contenido en una lista
5. Escribir un programa en Python lea el contenido de un archivo e identifique la palabra más larga de todo el archivo
6. Modificar el programa anterior para que imprima la frecuencia de repetición de cada palabra en el archivo
7. Escribir un programa en Python que lea información de texto de cuatro archivos distintos, la imprima por pantalla y luego la guarde en un archivo distinto.

9. Archivos JSON

1. Escribir un programa en Python que permita convertir tipos de datos de Python a JSON
2. Agregar al programa anterior el camino inverso
3. Escribir un programa en Python que permita leer información de un archivo json y mostrarla por pantalla de forma ordenada
4. Escribir un programa en Python que permita leer información de un tres json, las combine y lo almacene en toda en un cuarto archivo

10. Programación orientada a objetos

1. Crear una clase vehículo que contenga los atributos marca, velocidad máxima y kilómetros recorridos. La marca debe ser ingresada en el constructor y los demás atributos deben tener sus métodos getters y setters.
2. Agregar a la clase anterior agregar un método que permita comparar dos clases vehículo entre si. El atributo de comparación es la velocidad.
   También se debe agregar el método $\_\_str\_\_$ para que imprima información de la clase
3. Agregar infraestructura unit test al ejercicio anterior y comparar que la comparación funciona para vehículos más lentos, más rápidos o igual de veloces.

4. Definir una clase abstracta llamada vehículo con las interfaces arrancar(), frenar() y tiempo_de_marcha() Luego implementar las clases hijas:

   a) Colectivo

   b) Motocicleta

   c) Camión

   d) Automóvil

   Implementar cada uno de los métodos heredados de la clase abstracta. Luego, crear una lista con 10 instancias Colectivo, 2 motocicleta, 3 camiones y 3 automóviles. Se debe recorrer la lista, invocar a los métodos arrancar y esperar un tiempo random.
   Finalmente, se debe volver a recorrer la lista, invocando los métodos frenar e imprimir el tiempo de marcha de cada uno de los vehículos (diferencia de tiempos entre arrancar y frenar).