Preguntas teóricas:

¿Cuáles son los tipos de Datos en Python?

Los principales tipos de datos en Python son los siguientes:

- Cadenas: cadena de caracteres alfanuméricos, se redacta entre comillas dobles (") o simples (').
- Números: se redacta escribiendo el número sin más. Los números pueden ser enteros (28), flotantes (2.4) y complejos (3+4, por ejemplo).
- Listas: lista que contiene diferentes valores dentro de ella; se redacta entre corchetes y se divide el contenido con comas.
- Booleanos: tienen valor de "verdadero" o "falso".
- None: se utiliza para indicar que un valor todavía está sin definir.
- Tuplas: contienen varios elementos y se encierran entre paréntesis, no se pueden cambiar.
- Sets: conjunto de datos que no tienen un orden específico.
- Diccionarios: guarda pares de datos, una representación y su equivalente.
- Bytes y matrices de byte: permiten trabajar a nivel de bytes.

¿Qué tipo de convención de nomenclatura deberíamos utilizar para las variables en Python?

La convención de nomenclatura de variables es escribir cada palabra en minúsculas y unir las diferentes palabras que forman el nombre de una variable con barras bajas(_), a esto se la llama "*snake case*".
Además, se debe evitar utilizar ciertas letras solas como nombre de variable que puedan inducir a confusión: l (L minúscula), I (i mayúscula); O (o mayúscula), 0 (número).
Por último, los nombres de las variables deben ser descriptivos en relación con el contenido que guardan.

¿Qué es un Heredoc en Python?

Heredoc es el nombre que se le da a las cadenas formadas por varias líneas, como puede ser un texto con diferentes párrafos. Esta forma de encapsular un texto nos permite hacerlo de forma más sencilla que creando una cadena de una sola línea y haciendo las divisiones de forma manual con \n.
La manera de crear un Heredoc es añadiendo tres comillas (") al principio y otras tres al final, pegadas al texto o en líneas separadas. Hay que recordar que si ponemos las comillas en líneas propias, separadas del texto, se creará un espacio vacío antecediendo y precediendo el contenido.

¿Qué es una interpolación de cadenas?

La interpolación es el formateo de una cadena en la que es posible insertar el nombre de una variable entre llaves {} que, al imprimirlo, se sustituye por el valor de la variable. Hay dos formas de hacerlo, por un lado, crear la cadena antecedida por una *f* e insertando el nombre de la variable directamente; o, por otro lado, usando indexación dentro de la cadena y finalizarla con la función .format() donde el orden de los argumentos indicará a qué variable hace referencia cada uno.

¿Cuándo deberíamos usar comentarios en Python?

Por norma general, se deberían usar los comentarios para aquellas partes del programa que no van a cambiar en el tiempo, es decir, no se van a modificar cuando haya que actualizar alguna parte de la aplicación. Estos comentarios serán, por ejemplo, para indicar el contenido genérico de una página.
Como intentamos no introducir comentarios en todas partes, para que nuestro código se entienda con facilidad, deberíamos crear nombres de clases, variables, etc. que identifiquen el contenido con facilidad.

¿Cuáles son las diferencias entre aplicaciones monolíticas y de microservicios?

Las aplicaciones monolíticas están creadas en un mismo bloque, es decir, la totalidad de su estructura está contenida en una sola aplicación. Dentro de la aplicación, tendremos diferentes características. Los principales beneficios de este tipo de configuración son que se construyen de forma más rápida y no necesitan comunicarse de manera externa con otras características de la aplicación. En resumen, todo el contenido se encuentra en el mismo lugar. Lo malo de este tipo de arquitectura es que, si algo falla, puede hacer que toda la aplicación caiga. Además, son más difíciles de mantener y crear más características.
Por su parte, las aplicaciones de microservicios son aquellos que se crean de manera que cada característica forma su propio servidor y se conectan por medio de las API. El principal beneficio es que si una característica falla, no afecta al resto de componentes. Además, es más fácil desarrollar más componentes e identificar errores sin tener que cerrar la aplicación en su totalidad.