Checkpoint 3

¿Cuáles son los tipos de Datos en Python?
Hay 9 tipos de datos:

- Booleanos: True o False
- Números: 1, 2, 3
- Cadenas: ‘Kristine’, ‘Peter’
- Bytes y byte arrays
- Nulo
- Listas
- Tuplas
- Sets
- Diccionarios

Los últimos 4 forman parte de las colecciones.

¿Qué tipo de convención de nomenclatura deberíamos utilizar para las variables en Python?
Las variables se deben escribir en minúsculas y siendo lo más descriptivas posible. Si hay más de una palabra se separan por un guión bajo. No se deben utilizar mayúsculas, como Num_Two o NumTwo. Esta variable sería num_two. Tampoco se deben utilizar l (L minúscula) e I (i mayúscula) para nombrar una variable porque pueden confundirse.

¿Qué es un Heredoc en Python?
Un Heredoc es una cadena con múltiples párrafos o líneas de código. Se crea utilizando tres comillas al principio y al final del texto (“”” texto “”””) y sirve para introducir párrafos en el código. Se puede utilizar junto con la función .strip() para eliminar líneas en blanco al principio y al final del texto.

¿Qué es una interpolación de cadenas?
Consiste en introducir variables (cadenas, números) en lugares determinados dentro de otra cadena. Por ejemplo:

```
name = ‘Kristine’
greeting = f ‘Hi {name}’
```

En el caso anterior, donde pone name se escribirá el valor de la variable name, en este caso “Kristine”. Es muy útil para generar mensajes automáticos con los datos del usuario, por ejemplo.

¿Cuándo deberíamos usar comentarios en Python?
No se deben utilizar muchos comentarios en Python, se utilizan sobre todo para destacar las partes importantes de la aplicación o para marcar que hay que realizar futuros cambios en una determinada parte. Un exceso de comentarios puede dar lugar a malentendidos si otra persona tiene que entender nuestro código.

¿Cuáles son las diferencias entre aplicaciones monolíticas y de microservicios?
En las aplicaciones monolíticas todo el funcionamiento del sistema se basa en una sola aplicación, mientras que en las aplicaciones de microservicios puede haber varios servicios que hagan que el sistema funcione. Además, las monolíticas se desarrollan más rápido y, en general, son más rápidas que los microservicios, mientras que las de microservicios

requieren más tiempo para su correcto funcionamiento, ya que hay que configurar cada servicio. Por último, las monolíticas pueden ser más difíciles de actualizar porque un cambio puede estropear todo el sistema, mientras que en las de microservicios se pueden actualizar los servicios individualmente, por lo que si un componente falla no tiene por qué fallar toda la aplicación.