### Comprobación de errores

Deberíamos poder ejecutar el código *sitka\_highs\_lows.py* con los datos de cualquier lugar. Pero algunas estaciones meteorológicas recogen datos distintos de otras, y algunas funcionan mal ocasionalmente y no recogen algunos de los datos que se supone que deben recoger. La falta de datos puede dar lugar a excepciones que bloqueen nuestros programas, a menos que las manejemos adecuadamente.

Por ejemplo, veamos qué ocurre cuando intentamos generar un gráfico de temperatura del Valle de la Muerte, California. Copia el archivo *death\_valley\_2021\_simple.csv* en la carpeta donde guardes los datos para los programas de este capítulo.

Primero, vamos a ejecutar el código para ver las cabeceras que se incluyen en este archivo de datos:

**death\_valley\_highs\_lows.py**

from pathlib import Path  
import csv  
  
path = Path('weather\_data/death\_valley\_2021\_simple.csv')  
lines = path.read\_text().splitlines()  
  
reader = csv.reader(lines)  
header\_row = next(reader)  
  
for index, column\_header in enumerate(header\_row):  
 print(index, column\_header)

Éste es el resultado:

0 STATION  
1 NAME  
2 DATE  
3 TMAX  
4 TMIN  
5 TOBS

La fecha está en la misma posición, en el índice 2. Pero las temperaturas alta y baja están en los índices 3 y 4, por lo que tendremos que cambiar los índices en nuestro código para reflejar estas nuevas posiciones. En lugar de incluir una lectura de la temperatura media del día, esta estación incluye TOBS, una lectura para una hora de observación concreta.

Cambia *sitka\_highs\_lows.py* para generar un gráfico para el Valle de la Muerte utilizando los índices que acabamos de anotar, y observa qué ocurre:

**death\_valley\_highs\_lows.py**

--snip--  
path = Path('weather\_data/death\_valley\_2021\_simple.csv')  
lines = path.read\_text().splitlines()  
 --snip--  
# Extract dates, and high and low temperatures.  
dates, highs, lows = [], [], []  
for row in reader:  
 current\_date = datetime.strptime(row[2], '%Y-%m-%d')  
 high = int(row[3])  
 low = int(row[4])  
 dates.append(current\_date)  
--snip--

Actualizamos el programa para que lea del archivo de datos del Valle de la Muerte, y cambiamos los índices para que correspondan a las posiciones TMAX y TMIN de este archivo.

Cuando ejecutamos el programa, obtenemos un error:

Traceback (most recent call last):  
 File "death\_valley\_highs\_lows.py", line 17, in <module>  
 high = int(row[3])  
❶ ValueError: invalid literal for int() with base 10: ''

El rastreo nos dice que Python no puede procesar la temperatura alta de una de las fechas porque no puede convertir una cadena vacía ('') en un entero ❶. En lugar de buscar entre los datos para averiguar qué lectura falta, trataremos directamente los casos de datos que faltan.

Ejecutaremos código de comprobación de errores cuando se estén leyendo los valores del archivo CSV para gestionar las excepciones que puedan surgir. He aquí cómo hacerlo:

**death\_valley\_highs\_lows.py**

--snip--  
for row in reader:  
 current\_date = datetime.strptime(row[2], '%Y-%m-%d')  
❶ try:  
 high = int(row[3])  
 low = int(row[4])  
 except ValueError:  
❷ print(f"Missing data for {current\_date}")  
❸ else:  
 dates.append(current\_date)  
 highs.append(high)  
 lows.append(low)  
  
# Plot the high and low temperatures.  
--snip--  
  
# Format plot.  
❹ title = "Daily High and Low Temperatures, 2021\nDeath Valley, CA"  
ax.set\_title(title, fontsize=20)  
ax.set\_xlabel('', fontsize=16)  
--snip--

Cada vez que examinamos una fila, intentamos extraer la fecha y la temperatura alta y baja ❶. Si falta algún dato, Python lanzará un ValueError y lo manejamos imprimiendo un mensaje de error que incluye la fecha del dato que falta ❷. Tras imprimir el error, el bucle continuará procesando la siguiente fila. Si todos los datos de una fecha se recuperan sin error, se ejecutará el bloque else y los datos se añadirán a las listas correspondientes ❸. Como estamos trazando la información de una nueva ubicación, actualizamos el título para incluir la ubicación en el gráfico, y utilizamos un tamaño de letra más pequeño para acomodar el título más largo ❹.

Cuando ejecutes ahora *death\_valley\_highs\_lows.py*, verás que sólo faltaban datos de una fecha:

Missing data for 2021-05-04 00:00:00

Como el error se gestiona adecuadamente, nuestro código puede generar un gráfico que omite los datos que faltan. [La Figura 16-6](#figure16-6) muestra el gráfico resultante.

Comparando este gráfico con el de Sitka, podemos ver que el Valle de la Muerte es más cálido en general que el sureste de Alaska, como esperábamos. Además, el rango de temperaturas de cada día es mayor en el desierto. La altura de la región sombreada lo pone de manifiesto.

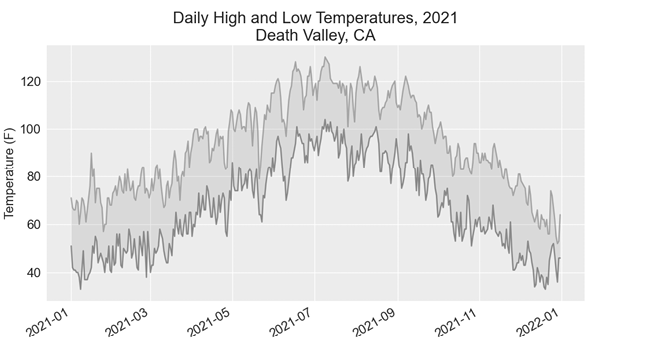


Figura 16-6: Temperaturas máximas y mínimas diarias

del Valle de la Muerte

Muchos conjuntos de datos con los que trabajes tendrán datos ausentes, mal formateados o incorrectos. Puedes utilizar las herramientas que aprendiste en la primera mitad de este libro para manejar estas situaciones. Aquí utilizamos un bloque try-except-else para gestionar los datos que faltan. A veces utilizarás continue para saltarte algunos datos, o utilizarás remove() o del para eliminar algunos datos después de haberlos extraído. Utiliza cualquier método que funcione, siempre que el resultado sea una visualización significativa y precisa.

[anterior](c16_12.html)[Subtema 13 de 29: (Ver todo)](c16.html)[siguiente](c16_14.html)