Capítulo 3: Describir datos con estadísticas

### **Retos de programación**

A continuación, aplica lo que has aprendido para completar los siguientes retos de programación.

#### ***#nº 1: Mejor programa de búsqueda de coeficientes de correlación***

La función find\_corr\_x\_y() que escribimos antes para hallar el coeficiente de correlación entre dos conjuntos de números supone que los dos conjuntos de números tienen la misma longitud. Mejora la función para que compruebe primero la longitud de las listas. Si son iguales, sólo entonces la función debe proceder con los cálculos restantes; de lo contrario, debe imprimir un mensaje de error indicando que no se puede encontrar la correlación.

#### ***#2: Calculadora estadística***

Implementa una calculadora estadística que tome una lista de números del archivo *misdatos.txt* y luego calcule e imprima su media, mediana, moda, varianza y desviación típica utilizando las funciones que escribimos anteriormente en este capítulo.

#### ***#3: Experimenta con otros datos CSV***

Puedes experimentar con numerosas fuentes de datos interesantes disponibles gratuitamente en Internet. El sitio web [*http://www.quandl.com/*](http://www.quandl.com/) es una de esas fuentes. Para este reto, descarga los siguientes datos como archivo CVS de [*http://www.quandl.com/WORLDBANK/USA\_SP\_POP\_TOTL/:*](http://www.quandl.com/WORLDBANK/USA_SP_POP_TOTL/) la población total de Estados Unidos al final de cada año para los años 1960 a 2012. A continuación, calcula la media, la mediana, la varianza y la desviación típica de la *diferencia* de población a lo largo de los años y crea un gráfico que muestre estas diferencias.

#### ***#nº 4: Hallar el percentil***

El percentil es una estadística de uso común que transmite el valor por debajo del cual cae un determinado porcentaje de observaciones. Por ejemplo, si un alumno obtuvo una puntuación percentil 95 en un examen, significa que el 95 por ciento de los alumnos obtuvieron una puntuación inferior o igual a la suya. Otro ejemplo: en la lista de números 5, 1, 9, 3, 14, 9 y 7, el percentil 50 es 7 y el percentil 25 es 3,5, un número que no está presente en la lista.

Hay varias formas de encontrar la observación correspondiente a un percentil determinado, pero aquí tienes un enfoque[.2](footnote.html#fn02)

Supongamos que queremos calcular la observación en el percentil *p*:

1. En orden ascendente, ordena la lista dada de números, que podríamos llamar data.

2. Calcula

image

donde *n* es el número de elementos de data.

3. Si *i* es un número entero, data[i] es el número correspondiente al percentil *p*.

4. Si *i* *no* es un número entero, establece *k* igual a la parte integral de *i* y *f* igual a la parte fraccionaria de *i*. El número (1-f)\*data[k] + f\*data[k+1] es el número correspondiente al percentil *p*.

Utilizando este enfoque, escribe un programa que tome un conjunto de números en un archivo y muestre el número que corresponde a un percentil específico suministrado como entrada al programa.

#### ***#5: Creación de una tabla de frecuencias agrupadas***

Para este reto, tu tarea consiste en escribir un programa que cree una tabla de frecuencias agrupadas a partir de un conjunto de números. Una tabla de frecuencias agrupadas muestra la frecuencia de los datos clasificados en diferentes *clases*. Por ejemplo, consideremos las puntuaciones de las que hablamos en "[Crear una](ch03.html#ch03lev2sec03)tabla de frecuencias" en [la página 69](ch03.html#page_69): 7, 8, 9, 2, 10, 9, 9, 9, 9, 4, 5, 6, 1, 5, 6, 7, 8, 6, 1 y 10. Una tabla de frecuencias agrupadas mostraría estos datos de la siguiente manera:

| **Curso** | **Frecuencia** |
| --- | --- |
| 1-6 | 6 |
| 6-11 | 14 |

La tabla clasifica los grados en dos clases: 1-6 (que incluye 1 pero no 6) y 6-11 (que incluye 6 pero no 11). Muestra frente a ellas el número de grados que pertenecen a cada categoría. Determinar el número de clases y el rango de números de cada clase son dos pasos clave para crear esta tabla. En este ejemplo, he mostrado dos clases con el rango de números de cada clase dividido a partes iguales entre las dos.

He aquí un enfoque sencillo para crear clases, que supone que el número de clases puede elegirse arbitrariamente:

def create\_classes(numbers, n):  
low = min(numbers)  
high = max(numbers)  
  
# Width of each class  
width = (high - low)/n  
classes = []  
a = low  
b = low + width  
classes = []  
while a < (high-width):  
classes.append((a, b))  
a = b  
b = a + width  
# The last class may be of a size that is less than width  
classes.append((a, high+1))  
return classes

La función create\_classes() acepta dos argumentos: una lista de números, numbers, y n, el número de clases a crear. Devolverá una lista de tuplas en la que cada tupla representa una clase. Por ejemplo, si se llama con los números 7, 8, 9, 2, 10, 9, 9, 9, 9, 4, 5, 6, 1, 5, 6, 7, 8, 6, 1, 10, y n = 4, devuelve la siguiente lista: [(1, 3.25), (3.25, 5.5), (5.5, 7.75), (7.75, 11)]. Una vez que tengas la lista, el siguiente paso es repasar cada uno de los números y averiguar a qué clase de las devueltas pertenece.

Tu reto consiste en escribir un programa que lea una lista de números de un fichero y luego imprima la tabla de frecuencias agrupadas, haciendo uso de la función create\_classes().

[anterior](ch03_9.html)[Subtema 10 de 10: (Ver todo)](ch03.html)