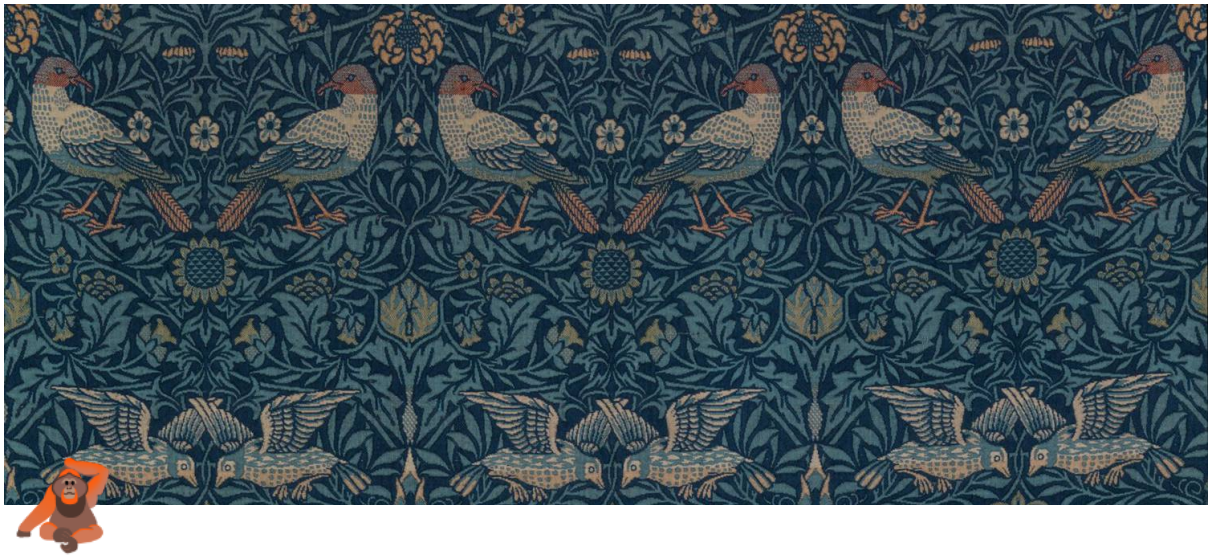
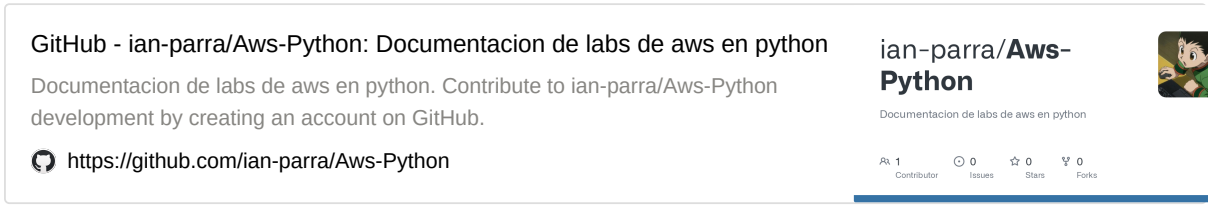


122- [PF] - Lab - Cálculo de la carga neta de la insulina mediante listas y bucles



Información general sobre el laboratorio

En el módulo de Control de caudal, aprendió sobre las instrucciones `if-else`, los bucles `while`, las listas y los bucles `for`. Ahora pondrá en práctica lo aprendido en la aplicación de la insulina en el mundo real.

Aquí utilizará bucles `lists`, `for` y `while`, y cálculos básicos para determinar la carga neta de la insulina entre pH 0 y pH 14.

En este laboratorio, deberá realizar lo siguiente:

- crear un diccionario de valores de pKa (que indican la intensidad de un ácido) que se utilizará en los cálculos de carga neta
- utilizar el método `count()` para obtener un recuento de aminoácidos
- utilizar un bucle `while` para calcular la carga neta de la insulina entre pH 0 y pH 14

Ejercicio 1: Asignación de variables, listas y diccionarios

1. En el panel de navegación del IDE, elija el archivo que creó en la sección *Creación del archivo de ejercicios de Python* anterior.
2. Copie el siguiente código, péguelo en el archivo y guarde el archivo:

```
# Python3.6
# Coding: utf-8
# Store the human preproinsulin sequence in a variable called preproinsulin:
preproInsulin = "malwmrllpllallalwgpdpaaafvnqhlcgshlvealylvcgergffytpktrreaedlqvgqvelgggpgagslqpl
alegslqkrgiveqcctsicslyqlenycn"
# Store the remaining sequence elements of human insulin in variables:
lsInsulin = "malwmrllpllallalwgpdpaaa"
bInsulin = "fvnqhlcgshlvealylvcgergffytpkt"
aInsulin = "giveqcctsicslyqlenycn"
cInsulin = "rreaedlqvgqvelgggpgagslqplalegslqkr"
insulin = bInsulin + aInsulin
```

1. En la línea siguiente, cree un diccionario nuevo y escriba: `pKR = {}`.
2. Para rellenar el diccionario con pares de clave-valor, inserte la primera clave de *y* con un valor de *10.07*. Coloque el cursor dentro de las llaves y escriba lo siguiente: `'y': 10.07,`.

> Nota: Se incluyó una coma después del valor para que pueda agregar los pares de clave-valor restantes.

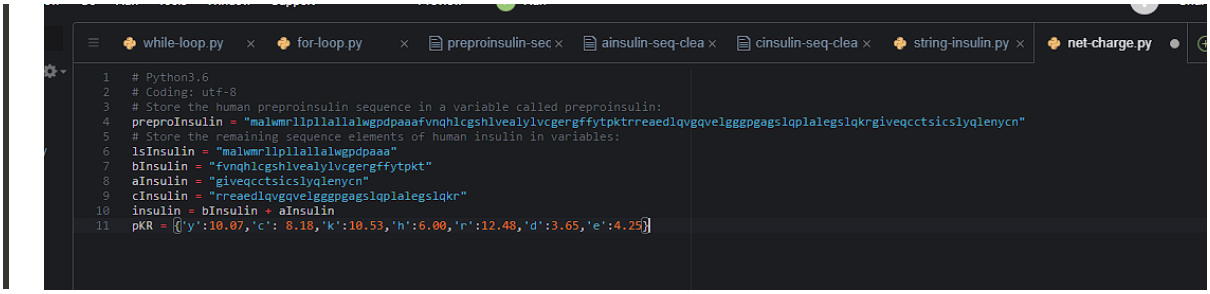
1. Agregue los siguientes pares de clave-valor al diccionario para que coincidan con el segmento de código.

- `'c': 8.18`
- `'k': 10.53`
- `'h': 6.00`
- `'r': 12.48`
- `'d': 3.65`
- `'e': 4.25`

El diccionario debería tener el mismo aspecto que el siguiente código:

```
pKR = {'y':10.07,'c': 8.18,'k':10.53,'h':6.00,'r':12.48,'d':3.65,'e':4.25}
```

> Nota: Y, C, K, H, R, D y E son los únicos aminoácidos que contribuyen al cálculo de la carga neta.



Ejercicio 2: Uso de count() para contar los números de cada aminoácido

En este ejercicio, utilizará el método `count()` y la comprensión de listas para contar la cantidad de aminoácidos Y, C, K, H, R, D y E. Estos aminoácidos contribuyen a la carga neta.

1. Para identificar cuántos elementos de un tipo hay dentro de una lista, puede utilizar el método `count()`. Para determinar cuántos aminoácidos *Y* hay en la insulina, utilice el método `count()` y escriba: `insulin.count("Y")`.
2. A continuación, actualice la línea `insulin.count()` al convertir la variable devuelta por el método `count()` como un flotante: `float(insulin.count("Y"))`.
3. Ahora que tiene la base para identificar una sola entidad, puede utilizar este método para buscar todas las entidades de una lista. Este proceso se puede realizar mediante la comprensión de listas. Para toda la línea, ingrese lo siguiente: `seqCount = ({x: float(insulin.count(x)) for x in ['y','c','k','h','r','d','e']})`

> Nota: Los dos primeros pasos de este ejercicio son los predecesores del tercer paso.

Ejercicio 3: Escritura de la fórmula de la carga neta

En este ejercicio, creará la fórmula de la carga neta. Utilizará la variable netCharge proporcionada en una fórmula de carga neta basada en Python. La función para la fórmula incluye un bucle `while` que imprimirá la carga neta mientras que la variable de pH es igual o inferior a 14.

1. Cree una variable llamada *pH* e iníciela desde cero, escriba `pH = 0` y presione ENTER (Intro).
2. Cree el bucle `while`, escriba `while (pH <= 14):` y presione ENTER (Intro).
3. Copie la siguiente variable *netCharge* y péguela al comienzo del bucle `while`.

```
netCharge = (
    +(sum({x: ((seqCount[x]*(10**pKR[x]))/((10**pH)+(10**pKR[x]))) \
    for x in ['k','h','r']}.values()))
```

```
    -(sum({x: ((seqCount[x]*(10**pH))/((10**pH)+(10**pKR[x]))) \
    for x in ['y','c','d','e']}.values())))
```

1. Para imprimir la variable *netCharge* con el *pH*, utilice una cadena de formato a fin de mejorar la legibilidad. Escriba `print('{0:.2f}'.format(pH), netCharge)` y presione ENTER (Intro).
2. Por último, incremente la variable de *pH*, escriba `pH +=1` y presione ENTER (Intro).
3. Guarde y ejecute el archivo.

Tenga cuidado con las sangrías y los espacios en Python.

```
# Python3.6
# Coding: utf-8
# Store the human preproinsulin sequence in a variable called preproinsulin:
preproInsulin = "malwmrllpllallalwgpdpaaafvnqhlcgshlvealylvcgergffytpktrreaedlqvgqvelgggpgagslqplalegslqkrgiveqcctsicslyqlenycn"
# Store the remaining sequence elements of human insulin in variables:
lsInsulin = "malwmrllpllallalwgpdpaaa"
bInsulin = "fvnqhlcgshlvealylvcgergffytpkt"
aInsulin = "giveqcctsicslyqlenycn"
cInsulin = "rreaedlqvgqvelgggpgagslqplalegslqkr"
insulin = bInsulin + aInsulin
pKR = {'y':10.07,'c': 8.18,'k':10.53,'h':6.00,'r':12.48,'d':3.65,'e':4.25}

seqCount = ({x: float(insulin.count(x)) for x in ['y','c','k','h','r','d','e']})
print(seqCount)

pH = 0
while (pH <= 14):
    netCharge = (
    +(sum({x: ((seqCount[x]*(10**pKR[x]))/((10**pH)+(10**pKR[x]))) \
    for x in ['k','h','r']}.values()))
    -(sum({x: ((seqCount[x]*(10**pH))/((10**pH)+(10**pKR[x]))) \
    for x in ['y','c','d','e']}.values()))
    print('{0:.2f}'.format(pH), netCharge)
    pH +=1
```

Los subconjuntos del código Python están organizados por sangrías y espacios. En Python, incluso una sangría o un espacio fuera de lugar puede generar una excepción u otro error. Por ejemplo, asegúrese de que cada elemento dentro de su bucle `while` tenga la sangría adecuada para que el código funcione.

¡Felicitaciones! Ha trabajado con listas y bucles en una función de Python.