Calculating the Net Charge of Insulin by Using Python Lists and Loops

Lab overview

In the Flow Control module, you learned about `if-else` statements, `while` loops, lists, and `for` loops. Now you will apply what you have learned to the real-world application of human insulin.

Here, you will use `lists` , `for` and `while` loops, and basic math to calculate the net charge of insulin from pH 0 to pH 14.

In this lab, you will:

- Create a dictionary of pKa values (which indicate the strength of an acid) that will be used in the net charge calculations
- Use the `count()` method to get a count of amino acids
- Use a `while` loop to calculate the net charge of insulin from pH 0 to pH 14

Estimated completion time

25 minutes

Accessing the AWS Cloud9 IDE

1. Start your lab environment by going to the top of these instructions and choosing **Start Lab**.

   A **Start Lab** panel opens, displaying the lab status.

2. Wait until you see the message *Lab status: ready*, and then close the **Start Lab** panel by choosing the **X**.

3. At the top of these instructions, choose **AWS**.

   The AWS Management Console opens in a new browser tab. The system automatically logs you in.

   > **Note:** If a new browser tab does not open, a banner or icon at the top of your browser typically indicates that your browser is preventing the site from opening pop-up windows. Choose the banner or icon, and choose **Allow pop ups**.

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
17

4. In the AWS Management Console, choose **Services** > **Cloud9**. In the **Your environments** panel, locate the **reStart-python-cloud9** card, and choose **Open IDE**.

   The AWS Cloud9 environment opens.

   > **Note:** If a pop-up window opens with the message *.c9/project.settings have been changed on disk*, choose **Discard** to ignore it. Likewise, if a dialog window prompts you to *Show third-party content*, choose **No** to decline.

Creating your Python exercise file

5. From the menu bar, choose **File > New From Template > Python File**.

   This action creates an untitled file.

6. Delete the sample code from the template file.
7. Choose **File > Save As...**, and provide a suitable name for the exercise file (for example, *net-charge.py*) and save it under the **/home/ec2-user/environment** directory.

Accessing the terminal session

8. In your AWS Cloud9 IDE, choose the **+** icon and select **New Terminal**.

   A terminal session opens.

9. To display the present working directory, enter `pwd` . This command points to **/home/ec2-user/environment**.
10. In this directory, you should also be able to locate the file you created in the previous section.

Exercise 1: Assigning variables, lists, and dictionaries

11. From the navigation pane of the IDE, choose the file that you created in the previous *Creating your Python exercise file* section.
12. Copy the following code, paste it into the file, and save the file:

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

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17

13. On the next line, create a new dictionary by entering: `pKR = {}`
14. To fill the dictionary with key-value pairs, insert the first key of *y* with a value of *10.07*. Place the cursor inside the braces, and enter: `'y': 10.07,`

    > **Note:** You included a comma after the value so that you can add the remaining key-value pairs.

15. To match the code segment, add the following key-value pairs into the dictionary.
    - `'c': 8.18`
    - `'k': 10.53`
    - `'h': 6.00`
    - `'r': 12.48`
    - `'d': 3.65`
    - `'e': 4.25`

    The dictionary should look like the following code:

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

    > **Note:** *Y*, *C*, *K*, *H*, *R*, *D*, and *E* are the only amino acids that contribute to the net-charge calculation.

Exercise 2: Using count() to count the numbers of each amino acid

In this exercise, you will use the `count()` method and list comprehension to count the number of Y, C, K, H, R, D, and E amino acids. These amino acids contribute to the net charge.

16. To identify a count of an item within a list, you can use the `count()` method. To see how many amino acids in insulin are *Y*, use the `count()` method by entering: `insulin.count("Y")`
17. Next, update the `insulin.count()` line by casting the variable returned by the `count()` method as a float: `float(insulin.count("Y"))`
18. Now that you have the basis for identifying a single entity, you can use this method to find all entities from a list. This process can be done by using list comprehension. For the entire line, enter: `seqCount = ({x: float(insulin.count(x)) for x in ['y','c','k','h','r','d','e']})`

    > **Note:** The first two steps in this exercise are predecessors to the third step.

Exercise 3: Writing the net charge formula

In this exercise, you will construct the net charge formula. You will use the provided netCharge variable in a Python-based net charge formula. The function for the formula includes a `while` loop that will print the net charge while the pH variable is equal to or below 14.

19. Create a variable called *pH* and initialize it to zero by entering `pH = 0` and pressing ENTER.
20. Create the `while` loop by entering `while (pH <= 14):` and pressing ENTER.
21. Copy the following *netCharge* variable and paste it at the beginning of the `while` loop.

```
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())))
```

22. To print the *netCharge* variable with the *pH*, use a format string for better readability. Enter `print('{0:.2f}'.format(pH), netCharge)` and press ENTER.
23. Finally, increment the *pH* variable by entering `pH +=1` and pressing ENTER.
24. Save and run the file.

Be careful about indentation and spacing in Python

Subsets of Python code are organized by indentation and spaces. In Python, even one misplaced indentation or space can throw an exception or other error. For example, be sure that every item within your `while` loop is properly indented so the code will work.

Congratulations! You have worked with lists and loops in a Python function.

End Lab

Congratulations! You have completed the lab.

25. Choose **End Lab** at the top of this page, and then select Yes to confirm that you want to end the lab.

    A panel indicates that *DELETE has been initiated... You may close this message box now.*

26. A message *Ended AWS Lab Successfully* is briefly displayed, indicating that the lab has ended.

Additional Resources

For more information about AWS Training and Certification, see https://aws.amazon.com/training/ (https://aws.amazon.com/training/).

Your feedback is welcome and appreciated. If you would like to share any suggestions or corrections, please provide the details in our AWS Training and Certification Contact Form (https://support.aws.amazon.com/#/contacts/aws-training).

- ○ Yes
- ○ No

< Rubric: 12 - Lists and Loops | Points: 0 >

Previous Next

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17