Aaron Eberly

CNIT 15501

Week 8 deliverable

Lists

A list is a way to store many different items under a single umbrella.

To create a list, put the items of the list within brackets. The following are two different ways to create an empty list:

cookies = []

cookies2 = list()

You can also create a list that already has items in it, as shown below:

cookies = ["plain", "chocolate chip", "sugar"]

3 strings are inside the above cookies list. However, a list could instead store other variables, such as ints and floats

Index

Each item on the list has an index starting at 0. For the above cookies list, “plain” is at index 0, “chocolate chip” is at index 1, and “sugar” is at index 2.

You can use brackets after a list name to choose an item at the specified index. For example, to print “chocolate chip”, you could use the following line:

print(cookies[1])

Remember that even though chocolate-chip is the second item in the list, it is at index 1.

You can use the len() function to return how many items are in a list. Still using the above list, here is an example.

X = len(cookies)

After this line of code, X will be 3. Notice how len() returns the number of items in the list, even though the index of the last item is only 2.

Say you wanted to print whatever item is last in the cookies list. You could use the following lines of code:

index = len(cookies) – 1

print(cookies[index])

When putting the index in brackets, you can enter a negative value. In this case, the index will count backwards, starting from the end of the list. The following is an example:

cookies = ["plain", "chocolate chip", "sugar"]

print(cookies[-1])

The print statement will print “sugar”.

If you attempt to enter an index larger than the length of the list -1, you will get an out-of-bounds error.

Adding Items to a List

I will use the following list for this section:

cookies = ["plain", "chocolate chip", "sugar"]

To add an item to the end of the list, you can use the append method.

cookies.append("oatmeal raisin")

The list now has 4 items: plain, chocolate chip, sugar, and oatmeal raisin.

If you want to add an item into the middle of the list, you can use insert. Insert has 2 parameters. The first is the index at which you want to add the new item. The second parameter is the item you want to add.

cookies = ["plain", "chocolate chip", "sugar"]

cookies.insert(1, "vanilla ice")

print(cookies)

The above code prints the following:

['plain', 'vanilla ice', 'chocolate chip', 'sugar']

Notice how Vanilla ice is now at index 1, pushing the later elements to a higher index.

Other List Methods

You can remove specific items from a list by using the remove method.

cookies.remove("plain")

This will remove plain from the list. If there were multiple “plain” items in the list, the remove function will only remove one of them.

You can also return the index of a specific item on a list. For example, if you wanted to discover the index of the item “sugar”, you could use the following lines of code:

cookies = ["plain", "chocolate chip", "sugar"]

pos = cookies.index("sugar")

print(pos)

The above code will print 2, because the index of sugar is 2.

Parallel lists

If you want multiple lists to contain items related to each other, you can do so by using index to relate the two lists. Here is an example:

cookies = ["plain", "chocolate chip", "sugar"]

calories = [100, 150, 120]

cookiePos = cookies.index("chocolate chip")

print(f"Chocolate chip cookies have {calories[cookiePos]} calories.")

The above code will print the following: Chocolate chip cookies have 150 calories.

Loops with Lists

You may use a for loop to reference each individual item in the list, one by one. Consider the following code:

cookies = ["plain", "chocolate chip", "sugar"]

for cookie in cookies:

    print(cookie)

The above code will print plain, chocolate chip, and sugar, each on a different line.

You can also do the same with a while loop, as shown below:

i = 0

while i < len(cookies):

    print(cookies[i])

    i += 1

Similarities Between Strings and Lists

len() and [index] work on strings as well as lists!

randomStr = 'hello'

print(len(randomStr))

The above code would print 5 because there are 5 characters in the string.

Enumerate Objects

Enumerate objects allow you to return both the index and item in a list. Below is an example:

for index, cookie in enumerate(cookies):

    print(f"The {cookie} is at rank: {index + 1}")

The above code prints the following:

The plain is at rank: 1

The chocolate chip is at rank: 2

The sugar is at rank: 3

CRUD

C – create (adding new items)

R – read (viewing info on items)

U – update (changing items)

D – delete (removing items)

Many programs will require these four functions to interact with a list. Each of these four functions can be accomplished with the above information on lists.