Adamson University
College of Engineering
Computer Engineering Department

Linear Algebra

Laboratory Activity No. 1

Getting Acquainted with Python

Submitted by:
Sustento, Myke Alvin

Instructor:
Engr. Dylan Josh D. Lopez

September 28, 2000

I. Objectives

This laboratory activity aims to implement the principles and techniques of Python as a programming language specifically creating a list and modifying it using functions such as zip(), sort(), append(), and pop().

II. Methods

The practices of the activity consist of using the sort function and the append function to modify a list. Sort function, as the name suggests, sorts a list ascendingly by default using a given function to specify the sorting criteria [1] while the append() function is used to add an element at the end of the list [2]. The reasoning behind using the sort() function and append() function is to achieve the deliverables of the activity which is to sort and append a given list of parties with pokemon strings as the elements.

III. Results

```
#list of pokemon in party
party = ['Charmander', 'Pidgey', 'Sandshrew', 'Rattata', 'Abra']
#list of levels
levels = [15, 11, 18, 5, 14]

#zip function to combine party and levels into a list of tuples
for pokemon,level in zip(party, levels):
    print(pokemon, "at level", level)
#I made the code into 1 line by just putting the print besides the for loop
##since codes are read left to right.

Charmander at level 15
Pidgey at level 11
Sandshrew at level 18
Rattata at level 5
Abra at level 14
```

Figure 1 Graded Cell 1

Figure 1 shows the graded cell 1 of the laboratory activity. The given was two lists, party list and levels list. The party list consisted of string elements while the levels list consisted of integer elements. The objective of graded cell 1 was to create an output where the elements at the party list and levels list are combined. The output was created using a for loop with a zip() function. The zip() function was used to combine the elements of party and levels into a single list of tuples [3] and the for loop was used to print the created list of tuples. In order to make this into a single line of code for additional points, the student coder typed the for loop

statement first and then the print statement second in a single line since codes are read left to right.

```
reserves = [
    ('Onix',10),
    ('Slowpoke',18),
    ('Dialga', 2),
    ('Magikarp', 32),
    ('Feebas', 22),
    ('Swablu', 19),
    ('Regigigas', 3),
    ('Unown', 50)
] #List of tuples of reserves

#sort() function to sort the reserves by level in descending order
#Lambda function to create an anonymous function that sorts using the level
reserves.sort(reverse=True, key=lambda level: level[1]);
#List comprehension to remove the levels and loop with 3 elements
picks = [pick[0] for pick in reserves[:3]];
#print the picks
print(picks)
#I made this into a 1 line code by separating sort(),
##List comprehension, and print with a ; (semi-colon)
```

```
['Unown', 'Magikarp', 'Feebas']
```

Figure 2 Graded Cell 2 (using list comprehension)

```
# Figure 3 Graded Cell 2 without using list comprehension
reserves = [
    ('Onix',10),
    ('Slowpoke',18),
    ('Dialga', 2),
    ('Magikarp', 32),
    ('Feebas', 22),
    ('Swablu', 19),
    ('Regigigas', 3),
    ('Unown', 50)
]

reserves.sort(reverse=True, key=lambda level: level[1]);

picks = []
for pick in reserves[:3]:
    picks.append(pick[0])
print(picks)
```

```
['Unown', 'Magikarp', 'Feebas']
```

Figure 3 Graded Cell 2 (without list comprehension)

Figure 2 shows the 2nd graded cell in the laboratory activity. In this graded cell, The given was a list of tuples called reserves and the objective was to identify and sort the top 3 highest or lowest level Pokémon and store them in a variable called “picks”. The student coder picked to sort it in descending order. Sort() function was used to sort the reserves with parameter reverse and key [2]. The reverse parameter was used to sort the list in descending order and this can be achieved by putting reverse=True in the sort() function [2]. The key parameter was used to specify the sorting criteria which were by levels [2]. The function used in the key was a lambda function with level as the argument and level[1] as the expression [4]. After sorting the list, a list comprehension was created to append the 3 highest level Pokémon in the picks list. To create a one-liner code, list comprehension was used instead of the normal for loop. List comprehension is a way to transform a list into another list [5] which is how reserves became picks. Elements can be conditionally included in the new list [5] so I included only the top 3 highest levels in reserves using reserves[:3]. Figure 2 and figure 3 yields the same result.

```
def create_party(party,candidates):
    suggested_parties = party
    suggested_parties.append(candidates[0])
    print(suggested_parties)
    suggested_parties.pop()
    suggested_parties.append(candidates[1])
    print(suggested_parties)
    suggested_parties.pop()
    suggested_parties.append(candidates[2])
    return suggested_parties
```

```
create_party(party,picks)
```

```
['Charmander', 'Pidgey', 'Sandshrew', 'Rattata', 'Abra', 'Unown']
['Charmander', 'Pidgey', 'Sandshrew', 'Rattata', 'Abra', 'Magikarp']
['Charmander', 'Pidgey', 'Sandshrew', 'Rattata', 'Abra', 'Feebas']
```

Figure 4 Graded Cell 3

Figure 3 shows the graded cell 3 of the laboratory activity. This graded cell contained a create_party function with party and candidates as a parameter. The objective of this graded cell is to create 3 lists that shows the party from graded cell 1 combined with the output in graded cell 2. In order to do the objective, suggested_parties = party since the output should show all the elements in the party list. In order to add the elements from the picks variable in graded cell 2, append() function was used to add candidates in the suggested_parties and then use the print() function to show the first list. After printing, pop() function was used to remove the last element in the list. Pop() function removes the last element in the list [6]. Repeat this process for candidates[1] and candidates[2] but use return suggested_parties instead of print to show the list created using candidates[2]. In order to show the outputs, call the create_party function with party and pokemon as global variables.

IV. Conclusion

In my opinion, Python is simpler than C++ because of its easy to read syntax. Python is great for beginners because it almost feels like reading English instead of codes.

The functions that I have learned and used in this laboratory exercise are zip(), sort(), append(), and lambda function. The zip() function returns a zip object, which is an iterator of tuples where the first item in each passed iterator is paired together, and then the second item, etc [3]. The zip() function was used in this activity to combine party and levels in graded cell

1. Sort() function is used to sort the list ascending by default [1] but it can be changed by typing true reverse=True in the reverse parameter. You can also make a function to decide the sorting criteria [1]. The sort() function was used to sort the integer part of the tuple, also called levels, in graded cell 2. To specify the sorting criteria as the levels, the lambda function to create an anonymous function that takes level as an argument and level[1] as the expression. A lambda function can take any number of arguments, but can only have one expression [4]. The append() function appends an element to the end of the list[2]. The append() function was used in both graded cells 2 and 3 to add elements on their respective list. Lastly, pop()function was used to remove the last element in the list[6]. This was used in graded cell 3 to remove the element created from the last print().

The advantages of using python as a programming language is that its syntax is easy to read because its operators uses words instead of symbols such as ‘and’, ‘or’, and in. Python also has list comprehension that turns a for loop code into one line of code. The advantages of using jupyter is that it has cells where you can separately run lines of code in different cell and you can choose to make the cell a markdown or a code. I can’t think of any disadvantages for python as a programming language as I don’t know a lot of programming languages. However, The disadvantage of jupyter notebook you need to run every cell or restart the kernel to get the right output.

References

[1] "Python List sort() Method", W3schools.com, 2020. [Online]. Available: https://www.w3schools.com/python/ref_list_sort.asp. [Accessed: 26- Sep- 2020].

[2] "Python List append() Method", W3schools.com, 2020. [Online]. Available: https://www.w3schools.com/python/ref_list_append.asp. [Accessed: 26- Sep- 2020].

[3] "The Python zip() Function", Stack Abuse, 2020. [Online]. Available: https://stackabuse.com/the-python-zip-function/#:~:text=The%20zip()%20function%20is,by%20the%20zip()%20function. [Accessed: 26- Sep- 2020].

[4] "Python Lambda", W3schools.com, 2020. [Online]. Available: https://www.w3schools.com/python/python_lambda.asp. [Accessed: 26- Sep- 2020].

[5] T. Hunner, "Python List Comprehensions: Explained Visually - Trey Hunner", Treyhunner.com, 2020. [Online]. Available: https://treyhunner.com/2015/12/python-list-comprehensions-now-in-color/. [Accessed: 26- Sep- 2020].

[6] "Python List pop() Method", W3schools.com, 2020. [Online]. Available: https://www.w3schools.com/python/ref_list_pop.asp. [Accessed: 27- Sep- 2020].

Appendix

Github Link - https://github.com/Sus102/LA_Lab