# annand.module06lab01

August 13, 2023

## 1 Assignment 6

#### 1.0.1 Joseph Annand

#### 1.0.2 8/13/2023

### 1.1 Question 1

1) import the random library.
2) Use `random.seed(10)` to initialize a pseudorandom number generator.
3) Create a list of 50 random integers from 0 to 15. Call this list `int_list`.
4) Print the 10th and 30th elements of the list.

You will need to use list comprehension to do this. The syntax for list comprehension is: = `[<expression> for <item> in <iterable>]`. For this question your expression will be a randint generator from the random library and your iterable will be `range()`. Researh the documentation on how to use both functions.

[12]:
```
# your code here
import random
random.seed(10)

int_list = [random.randint(0,15) for x in range(50)]
print(int_list[9])
print(int_list[29])
```

```
1
7
```

### 1.2 Question 2

1) import the string library.
2) Create the string `az_upper` using `string.ascii_uppercase`. This is a single string of uppercase letters
3) Create a list of each individual letter from the string. To do this you will need to iterate over the string and append each letter to the an empty list. Call this list `az_list`.
4) Print the list.

You will need to use a for-loop for this. The syntax for this for-loop should be:

`for i in string>:    <list operation>`

[13]:
```
# your code here
import string

az_upper = string.ascii_uppercase
az_list = []

for i in az_upper:
    az_list.append(i)
print(az_list)
```

```
['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P',
'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']
```

### 1.3 Question 3

1) Create a set from 1 to 5. Call this `set_1`.
2) Create a set from int_list. Call this `set_2`.
3) Create a set by finding the `symmetric_difference()` of `set_1` and `set_2`. Call this `set_3`.
4) What is the length of all three sets?

[14]:
```
# your code here
set_1 = {1,2,3,4,5}
set_2 = set(int_list)

set_3 = set_1.symmetric_difference(set_2)

print(len(set_1))
print(len(set_2))

# Why is set_2 only 15 elements when int_list has 50 elements?
# Sets do not allow duplicate values so set_2 only contains one copy of each␣
 ↪unique value in int_list

print(len(set_3))
```

```
5
15
12
```

### 1.4 Question 4

1) Import default dict and set the default value to 'Not Present'. Call this dict_1.
2) Add `int_list`, `set_2`, and `set_3` to `dict_1` using the object names as the key names.

3) Create a new dictionary, `dict_2`, using curly bracket notation with `set_1` and `az_list` as the keys and values.

4) Invoke the default value of `dict_1` by trying to access the key `az_list`. Create a new set named `set_4` from the value of `dict_1['az_list']`. What is the lenght of the difference between `dict_2['az_list']` and 'set_4'?

5) Update `dict_2` with `dict_1`. Print the value of the key `az_list` from `dict_2`. What happened?

[15]:
```
# your code here
# Part 1
from collections import defaultdict

def def_value():
    return "Not Present"

dict_1 = defaultdict(def_value)

# Part 2
dict_1.update({'int_list': int_list, 'set_2': set_2, 'set_3': set_3})

# Part 3
dict_2 = {'set_1': set_1, 'az_list': az_list}

# Part 4
print(dict_1['az_list'])
set_4 = {dict_1['az_list']} # set(dict_1['az_list']) returns a set with all␣
 ↪unique characters in the string "Not Present"
print(len(dict_2['az_list']) - len(set_4))

# Part 5
dict_2.update(dict_1)
print(dict_2['az_list'])

# When dict_1['az_list'] is called in Part 4, a key-value pair of 'az_list':␣
 ↪"Not Present" is created in dict_1.
# When dict_2 is updated using dict_1, the 'az_list' value of dict_2 is replaced␣
 ↪with "Not Present" from dict_1
```

```
Not Present
25
Not Present
```

[ ]:
```

```