Assignment 6

August 12, 2023

1 Assignment 6

1.0.1 Benjamin Michaels

1.0.2 8/11/23

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.

[17]:

```python
# your code here

import random
```

[ ]:

```python
random.seed()
```

[ ]:

```python
print(random.randint(0, 15))
```

[ ]:

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

[18]:

```python
print(int_list[10])
```

```
8
```

[19]:

```python
print(int_list[30])
```

```
4
```

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>
```

[57]:

```
# your code here

import string
```

[ ]:

```
az_upper = string.ascii_uppercase
```

[ ]:

```
az_list = []
for letter in az_upper:
    az_list.append(letter)
''
```

[ ]:

```
print(az_list)
```

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?

[10]:

```
# your code here

set_1 = {1, 2, 3, 4, 5}
```

[11]:

```
set_2 = set(int_list)
```

[16]:

```
set_3 = set_1.symmetric_difference(set_2)
```

[15]:

```
len(set_1)
```

[15]:

```
5
```

[14]:

```
len(set_2)
```

[14]:

```
16
```

[13]:

```
len(set_3)
```

[13]:
```
11
```

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?

[21]:
```python
# your code here

from collections import defaultdict
```

[22]:
```python
def def_value():
    return "Not Present"
```

[23]:
```python
dict_1 = defaultdict(def_value)
```

[24]:
```python
dict_1["int_list"] = int_list
```

[25]:
```python
dict_1["set_2"] = set_2
```

[26]:
```python
dict_1["set_3"] = set_3
```

[27]:
```python
dict_2 = defaultdict(def_value)
```

[28]:
```python
dict_2["set_1"] = set_1
```

[29]:
```python
dict_2["az_list"] = az_list
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[29], line 1
----> 1 dict_2["az_list"] = az_list

NameError: name 'az_list' is not defined
```

[30]:
```python
print(dict_1.get("az_list"))
```

```
None
```

```
[31]: set_4 = dict_1.get("az_list")
```

```
[32]: len(dict_2['az_list'])
      len(set_4)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[32], line 2
      1 len(dict_2['az_list'])
----> 2 len(set_4)

TypeError: object of type 'NoneType' has no len()
```

```
[ ]: dict_2.update(dict_1)
```

```
[ ]: for az_list in dict_2.items():
         print(az_list)
     #turned the original az_list into a tuple and then printed every key and value
```