Practice exercises: Week 8 (Dictionary and Set)

1. Write a program that creates a dictionary to save student names (first name only) and scores received in project work. The code should prompt the user to get inputs for name and followed by project work scores until the user input for name or scores is 0. [Hint: name-key: score-value). Then print the dictionary values.

```
stu = {}
score = 1
while score!=0:
    name = input("First name:")
    score = int(input("project work scores:"))
    if score ==0:
        break
    else:
        stu.update({name:score})
print(stu.items())
```

2. The dictionary → **dicAreas** contains country names and their land size as key and value respectively. Test the commands given below by using it to observe the results.

```
dicAreas={'Russia':1707.5,'Canada':997.1,'China':960.1,'American':936.4,'Brazil':854.7}
for item in dicAreas:
    print(item)
print("***"*10)
for item in dicAreas.items():
    print(item)
print("***"*10)
for k,v in dicAreas.items():
    print(k,v)
print("***"*10)
for key in dicAreas.keys():
    print (key)
print("***"*10)
for key in dicAreas.keys():
    print(key,dicAreas[key])
print("***"*10)
```

Answer:

```
>>> %Run ex3.py
Russia
Canada
China
American
Brazil
******************************
('Russia', 1707.5)
('Canada', 997.1)
('China', 960.1)
('American', 936.4)
('Brazil', 854.7)
******************************
Russia 1707.5
Canada 997.1
China 960.1
American 936.4
Brazil 854.7
******************************
Russia
Canada
China
American
Brazil
******************************
Russia 1707.5
Canada 997.1
China 960.1
American 936.4
Brazil 854.7
******************************
```

3. Write code to display the **dicArea's** elements in *descending order* by country name(*key*). Similarly, update your code that display elements of **dictArea** in *ascending order* by land size(*value*).

```
dicAreas={'Russia':1707.5,'Canada':997.1,'China':960.1,'American':936.
4,'Brazil':854.7}
#sort by key
for key in sorted(dicAreas):
    print(key, dicAreas[key])
#sort by value
sorted_values = sorted(dicAreas.values()) # Sort the values
sorted_dict = {}
for i in sorted_values:
    for k in dicAreas.keys():
        if dicAreas[k] == i:
            sorted_dict[k] = dicAreas[k]
            break
print(sorted_dict)
```

4. Write a program that merge two dictionaries namely **dicArea** (already you used for question 2 and 3) and **dicCaptials** shown here. Then display the merged results.

```
dicCapitals={'Russia':'Mosca', 'Canada':'Ottawa',
             'China':'Beijing', 'American':'Washington',
             'Brazil':'Brasilia'}
```

Here *country name* is a key and its *capital city*, and *land size* is a value. The expected output will be

```
*(4)****
('Russia', [1707.5, 'Mosca'])
('Canada', [997.1, 'Ottawa'])
('China', [960.1, 'Beijing'])
('American', [936.4, 'Washington'])
('Brazil', [854.7, 'Brasilia'])
```

Answer:

```
dicAreas={'Russia':1707.5,'Canada':997.1,'China':960.1,'American':936.
4,'Brazil':854.7}
#(4)Merge dictionaries "dicAreas" and "dicCapitals". In merged
dictionary "dicCountries",
#     the key is country name and corresponding value is landarea and
capital.
print("*(4)****")
dicCapitals={'Russia':'Mosca', 'Canada':'Ottawa',
                    'China':'Beijing', 'American':'Washington',
                    'Brazil':'Brasilia'}
dicCountries={}
for key in dicAreas.keys():
    dicCountries[key]=[dicAreas[key],dicCapitals[key]]
for item in dicCountries.items():
    print(item)
```

5. Generate 20 numbers in random that are in between 0 and 20 and store them in a list. Then print its values with no duplicates via transferring to another data structure that accepts no duplicates in *ascending order*. The sample run is here. [Hint: since it is random the output may vary]

```
>>> %Run ex4.py
20 random integers between 0 and 20 generated:
[2, 8, 10, 3, 8, 8, 6, 2, 12, 7, 3, 2, 1, 11, 16, 5, 13, 12, 20, 19]
The numbers different from each other:
{1, 2, 3, 5, 6, 7, 8, 10, 11, 12, 13, 16, 19, 20}
>>> %Run ex4.py
20 random integers between 0 and 20 generated:
[5, 9, 17, 0, 0, 9, 15, 4, 10, 3, 4, 9, 1, 3, 0, 6, 3, 13, 4, 1]
The numbers different from each other:
{0, 1, 3, 4, 5, 6, 9, 10, 13, 15, 17}
```

Answer:

```
import random
ls=[]
for i in range(20):
    ls.append(random.randint(0,20))
s=set(ls)
print("20 random integers between 0 and 20 generated:")
print(ls)
print("The numbers different from each other:")
print(s)
```

6. The set given below contains *product name* and *sales* of product for the month in tuple format.

```
monset={("Mobile phone",71200),("Router",18200),
        ("Portable Hard Disk",12000),("Printer",25600)}
```

Compute the **total month sale**. The expected output will be:

```
>>> %Run ex5.py
Total sales is: 127000
```

Answer:

```
monset={("Mobile phone",71200),("Router",18200),
        ("Portable Hard Disk",12000),("Printer",25600)}
total=0
for item in monset:
    total+=item[1]
print("Total sales is:",total)
```

7. The sets A and B are given here. Implement set operations → *union, intersect, difference* and *symmetric_difference* for these sets via Python coding using **set** data structure.

```
A={1,2,3,4,5}
B={4,5,6,7,8}
```

The expected output will be:

```
>>> %Run ex6.py
Union:
{1, 2, 3, 4, 5, 6, 7, 8}
{1, 2, 3, 4, 5, 6, 7, 8}
Intersection:
{4, 5}
{4, 5}
Difference:
{1, 2, 3}
{1, 2, 3}
Symmetric_difference:
{1, 2, 3, 6, 7, 8}
{1, 2, 3, 6, 7, 8}
```

Answer:

```
A={1,2,3,4,5}
B={4,5,6,7,8}
print("Union:")
print(A|B)
print(A.union(B))
print("Intersection:")
print(A&B)
print(A.intersection(B))
print("Difference:")
print(A-B)
print(A.difference(B))
print("Symmetric_difference:")
print(A^B)
print(A.symmetric_difference(B))
```

8. The top 10 programming languages listed by IEEE spectrum are *python, Java, C, C++, JavaScript, C#, R, go, HTML* and *swift*. The top 10 programming languages are listed by TIOPE index in September 2021 are: *C, Python, Java, C++, C#, VB, JavaScript, ASM, PHP,* and *SQL*. These were represented as 2 sets:

```
setIEEE={'Python','Java','C','C++','JavaScript',
         'C#','R','Go','HTML','swift'}
setTIOPE={'C','Python','Java','C++','C#','VB',
          'JavaScript','ASM','PHP','SQL'}
```

Write code that print the results of the following:

(i) All Programming languages of IEEE and TIOPE
(ii) Programming languages that exist in both IEEE and TIOPE
(iii) Programming languages that exists in IEEE but not in TIOPE
(iv) Programming language that exists only in TIOPE but not in IEEE and languages that exists in IEEE but not in TIOPE-combined

The sample run is here: [student descrtion/interpreation is necessary]

```
>>> %Run ex7.py
  Programming languages in two top 10 lists:
  {'Java', 'VB', 'JavaScript', 'HTML', 'Python', 'swift', 'Go', 'C', 'ASM', 'SQL'
  , 'C++', 'PHP', 'R', 'C#'}
  Programming languages in both top 10 lists:
  {'C', 'Java', 'JavaScript', 'C++', 'Python', 'C#'}
  Programming languages only in IEEE top 10 list:
  {'R', 'swift', 'HTML', 'Go'}
  Programming languages only in one top 10 list:
  {'VB', 'ASM', 'HTML', 'SQL', 'PHP', 'R', 'swift', 'Go'}
```

Answer:

```
setIEEE={'Python','Java','C','C++','JavaScript',
         'C#','R','Go','HTML','swift'}
setTIOPE={'C','Python','Java','C++','C#','VB',
          'JavaScript','ASM','PHP','SQL'}
print("Programming languages in two top 10 lists:")
print(setIEEE|setTIOPE)
print("Programming languages in both top 10 lists:")
print(setIEEE & setTIOPE)
print("Programming languages only in IEEE top 10 list:")
print(setIEEE-setTIOPE)
print("Programming languages only in one top 10 list:")
print(setIEEE^setTIOPE)
```

9. Define a module named **AreaEx9** that contains 3 subprograms namely

(i) **info()** – provides information about the module like what it has..

(ii) **cirArea()** – accepts radius value as argument and return the computed area of the circle value to called program

(iii) **triArea()** - return the computed area of the trianlge.

Module: AreaEx8 (AreaEx8.py)

```
import math
pi=3.1415926
def info():
    print("This module include 2 functions:")
    print("cirArea() for circle area calculate.")
    print("triArea() for triangle area calculate.")
def cirArea():
    r=float(input("Input radius of circle:"))
    if r<0:
        print("please input r greater than 0.")
    else:
        s=pi*r*r
        print("Circle area is:",s)
def triArea():
    a=float(input("input a:"))
    b=float(input("input b:"))
    c=float(input("input c:"))
    if a+b>c and a+c>b and b+c>a:
        p=(a+b+c)/2
        s=math.sqrt(p*(p-a)*(p-b)*(p-c))
        print("The area of triangle is:",s)
    else:
        print("Can not form a triangle.")
```

Write a main program to test the module's functions/procedures created above.

```
import AreaEx8
AreaEx8.info()
AreaEx8.cirArea()
AreaEx8.triArea()
```