DSO-599 Practice Exam II (with Solutions)

Part I. Multiple Choice (1 point each)

1. Predict the result of the following expression

```
((3**2-5)>1)*2+len('45')
```

A) 2
B) 4
C) 47
D) 245
E) None of the above

```
[1]: ((3**2-5)>1)*2+len('45')
```

```
4
```

2. Predict the result of the following expression

```
{'5':3,'2':1}[str(2)]+[6,3][0]
```

A) 9
B) 8
C) 7
D) 4
E) 2

```
[2]: {'5':3,'2':1}[str(2)]+[6,3][0]
```

```
7
```

3. Predict the output of the following code

```
def f(x):
    if type(x)==str:
        return 1
    else:
        return x+3
print(f(f('5'))-f(5))
```

A) -4
B) -1
C) 0
D) 1
E) 3

```
[3]: def f(x):
         if type(x)==str:
             return 1
         else:
             return x+3
     print(f(f('5'))-f(5))
```

```
-4
```

4. Predict the output of the following code

```
a=4
s=0
while a>1:
    s+=a
    a-=2
print(s)
```

A) 0
B) 2
C) 4
D) 6
E) 9

```
[4]: a=4
     s=0
     while a>1:
         s+=a
         a-=2
     print(s)
```

```
6
```

5. Predict the output of the following code

```
dic={'3':0,'4':2,'5':1}
lis=[3,4,5]
x=lis [ dic [ str( lis[1] ) ] ]
print(x)
```

A) 1
B) 2
C) 3
D) 4
E) 5

```
[5]: dic={'3':0,'4':2,'5':1}
     lis=[3,4,5]
     x=lis [ dic [ str( lis[1] ) ] ]
     print(x)
```

```
5
```

6. Predict the result of the last line of the following code

```
import pandas as pd
s=pd.Series([0,2,4,6,8])
s[s<=4].mean()+(s<=4).sum()
```

A) 3.6
B) 4.0
C) 4.2
D) 5.0
E) 8.0

```
[6]: import pandas as pd
     s=pd.Series([0,2,4,6,8])
     s[s<=4].mean()+(s<=4).sum()
```

```
5.0
```

7. Predict the result of the last line of the following code

```
import pandas as pd
s=pd.Series(['apple','orange','watermelon'])
len(s.str[4:][2])
```

A) 0
B) 1
C) 2
D) 4
E) 6

```
[7]: import pandas as pd
     s=pd.Series(['apple','orange','watermelon'])
     len(s.str[4:][2])
```

```
6
```

8. Predict the output of the following code

```
import pandas as pd
s=pd.to_numeric(pd.Series(['apple',1,'orange',3]),errors='coerce')
s.isnull().sum()
```

A) NaN
B) 1
C) 2
D) 3
E) 4

```
[8]: import pandas as pd
     s=pd.to_numeric(pd.Series(['apple',1,'orange',3]),errors='coerce')
     s.isnull().sum()
```

```
2
```

9. Which of the following lines will NOT cause an error?

A) `[3,5](0)+1`
B) `{0:[0,0],1:[1,1]}[0][0]`
C) `str('3'+int(3.2))`
D) `[2,3].iloc[0]`
E) `int(' 352 ').strip()`

```
[9]: {0:[0,0],1:[1,1]}[0][0]
```

```
0
```

10. Which of the following lines will NOT cause an error?

Assume the following code is executed first:

```
import pandas as pd
s1=pd.Series([0,1,2,3])
s2=pd.Series([4,3,2,1])
```

A) `s1.iloc[1,1]`
B) `pd.DataFrame([s1,s2])['s1']`
C) `pd.DataFrame([s1,s2]).loc['s1',0]`
D) `s1[s1[0]+s2[0]]`
E) `(s1+(s2>2)).sum()`

```
[10]: import pandas as pd
      s1=pd.Series([0,1,2,3])
      s2=pd.Series([4,3,2,1])
      (s1+(s2>2)).sum()
```

```
8
```

Part II. Writing Code

11. Manipulating Data (3 points)

Given a Dataframe called "grades" as follows (the first column is called "name" and stores the name of the student, and all subsequent columns correspond to an assignment):

	name	hwk1	hwk2	hwk3
0	Alice	1.0	0.9	0.8
1	Bob	0.8	1.0	0.9
2	Charlie	1.0	0.0	1.0

Create a DataFrame called "grades2" that aggregates all the assignment data into two columns as below: (Note: your code must work given whatever data in "grades", even if there are more than 3 homeworks; in other words, you will not obtain points if you hard-code the answer.)

	name	assignment	points
0	Alice	hwk1	1.0

	name	assignment	points
1	Bob	hwk1	0.8
2	Charlie	hwk1	1.0
3	Alice	hwk2	0.9
4	Bob	hwk2	1.0
5	Charlie	hwk2	0.0
6	Alice	hwk3	0.8
7	Bob	hwk3	0.9
8	Charlie	hwk3	1.0

Complete the following code to make the DataFrame called "grades2":

```
import pandas as pd
grades=pd.DataFrame([['Alice',1,0.9,0.8],['Bob',0.8,1,0.9],['Charlie',1,0,1]],\
                    columns=['name','hwk1','hwk2','hwk3'])
```

```
[11]: import pandas as pd
      grades=pd.DataFrame([['Alice',1,0.9,0.8],['Bob',0.8,1,0.9],['Charlie',1,0,1]],\
                          columns=['name','hwk1','hwk2','hwk3'])
      grades2=grades.melt(id_vars='name',var_name='assignment',value_name='points')
```

12. Analyzing Data in List Format (5 points)

Write a function called "analyze" which takes two inputs:

- lis: a list of numbers.
- threshold (default value=0): a numerical threshold

The function should return the sum of the numbers in the list that are greater than or equal to the given threshold. You may assume that all items in "lis" are numbers and that the threshold is a valid number.

Example 1:

```
analyze([3,5,4],5)
```

yields the result:
5

Example 2:

```
analyze([3,5,4],6)
```

0

Example 3:

```
analyze([1,2,3,4])
```

10

```
[12]: def analyze(lis,threshold=0):
          s=0
          for value in lis:
              if value>=threshold:
                  s+=value
          return s
```

```
[13]: analyze([3,5,4],5)

5

[14]: analyze([3,5,4],6)

0

[15]: analyze([1,2,3,4])

10

[16]: # Alternative solution using Pandas
      import pandas as pd
      def analyze(lis,threshold=0):
          ser=pd.Series(lis)
          return ser[ser>=threshold].sum()
```

13. Interactive Number Cruncher (7 points)

Write a program which repeatedly reads numbers using the prompt "Enter a number:" until the user enters "done" (case sensitive). Once "done" is entered, print out the total, count, and average of the numbers. (You can assume that the user inputs at least one valid number before inputing done, so the count will never be zero.)

If the user enters anything other than a number, detect their mistake using try and except, and print the error message "Invalid input" and skip to the next number.

Sample run 1:

```
Enter a number: 4
Enter a number: 5
Enter a number: '4'
Invalid input
Enter a number: 6
Enter a number: done
15.0 3 5.0
```

Sample run 2:

```
Enter a number: -1
Enter a number: five
Invalid input
Enter a number: 3.6
Enter a number: DONE
Invalid input
Enter a number: done
2.6 2 1.3
```

Write your code below:

```
[17]: total=0
      count=0
      while True:
          userInput=input('Enter a number: ')
          if userInput=='done':
```

```
            break
        try:
            number=float(userInput)
        except:
            print('Invalid input')
            continue
        total+=number
        count+=1
    print(total,count,total/count)
```

```
Enter a number: 4
Enter a number: 5
Enter a number: '4'
Invalid input
Enter a number: 6
Enter a number: done
15.0 3 5.0
```