Programming Workshop 5 SOLUTIONS

Module 9

1. **Trace the output.**

```
 try:
      num1 = 4
      num2 = 0
      answer = num1 / num2

except:
      print("Cannot divide by zero.")

else:
      print(num1, "/", num2, "=", format(answer, '.2f'))
```

```
Cannot divide by zero.
```

Given the following file *grocery.txt*, trace the output of the program below.

```
apples
bananas
milk
cheese
orange juice
```

```
pointer = open("grocery.txt", "r")
data = pointer.read()
items = data.split("\n")

if "Milk" in items:
      print("This item is in your groceries list.")

else:
      print("This item is NOT in your groceries list.")
```

```
This item is NOT in your groceries list.
```

2. **True or False.**

Statement	True / False

`try:`
`  x = float('abc123')`
`  print('The conversion is complete.')`
`except:`
`  print('Something went wrong.')`
 	
`print('The end.')`
 	
This program will output `Something went wrong.`
	`False`
	
	`Something went wrong. The end.`
	
	`Else is optional`

`try:`
`  x = float('123')`
`  print('The conversion is complete.')`
`except:`
`  print('Something went wrong.')`
 	
`print('The end.')`
 	
This program will output :
`The conversion is complete.`
`The end.`
	`True`

`values = [2] * 5`
`print(values)`
 	
This program will output `[10]`
	`False`
	
	`[2,2,2,2,2]`

Lists in Python are immutable.	`True`
If a file already exists and we open the file in `"w"` mode, it will erase any data that was already written in the file.	`True`
In a `try` / `except` / `else` statement, the `else` block will always execute regardless if there is an error in the `try` block.	`False`

3. **Reading CSV Files**

Assume you are given the following file called *excel_gradedata.csv* (available on Text Files website). It contains a list of students and 3 test scores for each student.

The contents of *excel_gradedata.csv* looks like this:

```
Manes,Arnoldo,37,35,93
MacCorley,Garnette,67,63,68
Stelfax,Eldredge,93,60,51
Dillicate,Fee,34,93,50
Antonin,Cammy,61,84,78
Ziemsen,Barb,93,84,33
Pourveer,Shalna,88,60,38
Carletti,Mitchel,83,57,95
Cromar,Jordana,79,96,46
Lathey,Cati,95,79,100
```

Write a Python program that opens it up and prints out the average score for each student that is represented in the file.

Sample output:

```
Arnoldo's average: 55.00
Garnette's average: 66.00
Eldredge's average: 68.00
Fee's average: 59.00
Cammy's average: 74.33
Barb's average: 70.00
Shalna's average: 62.00
Mitchel's average: 78.33
Jordana's average: 73.67
Cati's average: 91.33
```

```
# open file
file = open("excel_gradedata.csv", "r")
data = file.read()
file.close()

# split data by line
lines = data.split("\n")

#look at each line
for l in lines:
    #print(l)
    #extract student data
    student_data = l.split(",")
    #print(student_data)

    #extract grades
    grades = student_data[2:]
    #print(grades)

    #calculate total (many ways to do this)
    total = 0
    for g in grades:
        total += int(g)
    average = total/len(grades)

    print(student_data[1] + "'s", "average:", format(average, '.2f'))
```

4. **World Series**

Assume you are given a called *worldSeries.txt* (available on Text Files website).This data file contains all World Series winning teams up until a few years ago.

Here is a snippet of what it contains:

```
Boston Americans
New York Giants
Chicago White Sox
Chicago Cubs
Chicago Cubs
Pittsburgh Pirates
Philadelphia Athletics
Philadelphia Athletics
Boston Red Sox
Philadelphia Athletics
Boston Braves
Boston Red Sox
Boston Red Sox
Chicago White Sox
Boston Red Sox
```

Write a program that reads in this data and finds the team that won the most games.

```
# open file
```

```
file = open('worldSeries.txt', 'r')
data = file.read()

# convert data string into list
teams = data.split("\n")
#print(teams)

# analyze frequency of teams
unique = []
count = []

for t in teams:
    if t not in unique:
        #add to list
        unique.append(t)
        count.append(1)
    else:
        #find where it is located and update count
        pos = unique.index(t)
        count[pos] += 1

#print(unique)
#print(count)

# find the team with most wins
wins = max(count)
posWins = count.index(wins)
winTeam = unique[posWins]
print("Winning team is", winTeam)
```

5. **Magic Ball**

Write a program that simulates a Magic 8 Ball, which is a fortune-telling toy that displays a random response to a yes or no question. For this program, use a file called *8_ball_responses.txt*. The file contains 12 responses, such as “I don’t think so”, “Yes, of course!”, “I’m not sure”, and so forth. The program should read the responses from the file and convert it into a list. It should prompt the user to ask a question, then display one of the responses, randomly selected from the list. The program should repeat until the user enters the word "end".

Contents of *8_ball_responses.txt*:

```
Yes, of course!
Without a doubt, yes.
You can count on it.
For sure!
Ask me later.
I'm not sure.
I can't tell you right now.
I'll tell you after my nap.
No way!
I don't think so.
Without a doubt, no.
The answer is clearly NO.
```

Sample output (note your answers might be different):

```
Enter your question. Will I be rich?
I'll tell you after my nap.
Enter your question. Will I pass this class?
For sure!
Enter your question. Can I make all my dreams come true?
The answer is clearly NO.
Enter your question. end
```

```
import random
```

```
# read file
f = open('8_ball_responses.txt', 'r')
data = f.read()
f.close()

# clean data

# extract responses and convert to a list
responses = data.split('\n')
#print(responses)

question = input("Enter your question. ")

while question != 'end':

    # pick a response
    position = random.randint(0, len(responses)-1)
    print(responses[position])

    # ask again
    question = input("Enter your question. ")
```

ASCII Code Table

0	<NUL>	32	<SPC>	64	@	96	`	128	Ä	160	†	192	¿	224	‡
1	<SOH>	33	!	65	A	97	a	129	Å	161	°	193	¡	225	·
2	<STX>	34	"	66	B	98	b	130	Ç	162	¢	194	¬	226	‚
3	<ETX>	35	#	67	C	99	c	131	É	163	£	195	√	227	„
4	<EOT>	36	$	68	D	100	d	132	Ñ	164	§	196	ƒ	228	‰
5	<ENQ>	37	%	69	E	101	e	133	Ö	165	•	197	≈	229	Â
6	<ACK>	38	&	70	F	102	f	134	Ü	166	¶	198	∆	230	Ê
7	<BEL>	39	'	71	G	103	g	135	á	167	ß	199	«	231	Á
8	<BS>	40	(	72	H	104	h	136	à	168	®	200	»	232	Ë
9	<TAB>	41	)	73	I	105	i	137	â	169	©	201	…	233	È
10	<LF>	42	*	74	J	106	j	138	ä	170	™	202		234	Í
11	<VT>	43	+	75	K	107	k	139	ã	171	´	203	À	235	Î
12	<FF>	44	,	76	L	108	l	140	å	172	¨	204	Ã	236	Ï
13	<CR>	45	-	77	M	109	m	141	ç	173	≠	205	Õ	237	Ì
14	<SO>	46	.	78	N	110	n	142	é	174	Æ	206	Œ	238	Ó
15	<SI>	47	/	79	O	111	o	143	è	175	Ø	207	œ	239	Ô
16	<DLE>	48	0	80	P	112	p	144	ê	176	∞	208	–	240	
17	<DC1>	49	1	81	Q	113	q	145	ë	177	±	209	—	241	Ò
18	<DC2>	50	2	82	R	114	r	146	í	178	≤	210	“	242	Ú
19	<DC3>	51	3	83	S	115	s	147	ì	179	≥	211	”	243	Û
20	<DC4>	52	4	84	T	116	t	148	î	180	¥	212	‘	244	Ù
21	<NAK>	53	5	85	U	117	u	149	ï	181	µ	213	’	245	ı
22	<SYN	54	6	86	V	118	v	150	ñ	182	∂	214	÷	246	ˆ
23	<ETB>	55	7	87	W	119	w	151	ó	183	∑	215	◊	247	˜
24	<CAN>	56	8	88	X	120	x	152	ò	184	∏	216	ÿ	248	¯
25	<EM>	57	9	89	Y	121	y	153	ô	185	π	217	Ÿ	249	˘
26	<SUB>	58	:	90	Z	122	z	154	ö	186	∫	218	⁄	250	˙
27	<ESC>	59	;	91	[	123	{	155	õ	187	ª	219	€	251	°
28	<FS>	60	<	92	\	124	|	156	ú	188	º	220	‹	252	¸
29	<GS>	61	=	93	]	125	}	157	ù	189	Ω	221	›	253	˝
30	<RS>	62	>	94	^	126	~	158	û	190	æ	222	fi	254	˛
31	<US>	63	?	95	_	127	<DEL>	159	ü	191	ø	223	fl	255	ˇ