

## CODES:

January 13, 2019

```
1 i=1
2 while i<=5
3     print(i)
4     i=i+1
5 print("Finished!")
```

Listing 1: 1-print.py

```
1 quotient=20//3;
2 remainder=16%5;
3 print(quotient);
4 print(remainder);
5 print(7%(5//2));
```

Listing 2: 1-print-quo-rem.py

```
1 words=["spam","egg","spam","sausage"]
2 print("spam" in words)
3 print("egg" in words)
```

Listing 3: agumending-list-and-repeat-list.py

```
1 print("Janani " + "loves " + "Elang")
```

Listing 4: ~~concatenation.py~~

```
1 def my_func():
2     print("Spam\n" * 3)
3
4 my_func()
```

Listing 5: def-func.py

```
1 ages={"Dave":24, "Mary":42, "Charl":58}
2 print(ages["Dave"])
3 print(ages["Mary"])
```

Listing 6: dictioaries.py

```
1 primes={1:2, 2:3, 4:7, 7:17 }
2 print(primes[4])
3 print(primes[primes[4]])
```

Listing 7: dictionaries-adding-element.py

```

1 try :
2     num1=7
3     num2=7
4     print(num1/num2)
5     print("Done Calculation!")
6 except ZeroDivisionError:
7     print("An error occurred")
8     print("due to zero division")

```

Listing 8: exception-andling.py

```

1 file=open("testfile.txt","r")
2 print("Reading")
3 print(file.read())
4 print("Re-reading")
5 print(file.read())

```

Listing 9: file-rereading-will-return-nothing.py

```

1 try :
2     print("Hello")
3     print(1/0)
4 except ZeroDivisionError:
5     print("Dived by zero")
6 finally :
7     print("This code will run no matter what")

```

Listing 10: finally.py

```

1 print("2"+"3")
2 a=int("2")+int(3)
3 b=4+7
4 c=int(3.1)+int(5.8)
5 d="3"*2
6 print(a)
7 print(b)
8 print(c)
9 print(d)
10 e=int(d)
11 print(e)

```

Listing 11: float-to-int-type-conversion.py

```

1 def add(x,y):
2     return x+y
3
4 def do_twice(func,x,y):
5     return func(func(x,y),func(x,y))
6
7 a=5
8 b=6
9
10 print(do_twice(add,a,b))

```

Listing 12: fucn-euqals-variable.py

```

1 def shout(word):
2     """
3     Print a word with an exclamation mark following itself.
4     """
5     print(word+"!")
6
7 shout("ellai")

```

Listing 13: func-as-func-input.py

```

1 import random
2
3 for i in range(5):
4     value=random.randint(10,16)
5     print(value)

```

Listing 14: func-to-a-func.py

```

1 def even(x):
2     if x%2==0:
3         print("yes")
4     else:
5         print("No")
6
7 even(9)
8 even(6)

```

Listing 15: func-with-inp-arg.py

```

1 pairs={
2     1:"apple",
3     "orange":[2, 3, 4],
4     True:False,
5     None:"True",
6 }
7
8 print(pairs.get("orange"))
9 print(pairs.get(7))
10 print(pairs.get(1,"ae its there")) #this results in false
11 print(pairs.get(12345,"this is not in dict"))

```

Listing 16: get-in-dict.py

```

1 print(10/50)

```

Listing 17: hello-world.py

```

1 from math import pi,sqrt
2
3 print(pi)
4 print(sqrt(5))

```

Listing 18: import-module.py

```

1 from math import sqrt as square_root
2 print(square_root(3))

```

Listing 19: import-more-than-two-module.py

```

1 from math import sqrt as square_root
2 print(square_root(3))

```

Listing 20: imprt-in-diff-name.py

```

1 name=input("enter something:")
2 print("Nice to meet you "+name)

```

Listing 21: input-print.py

```

1 nums={
2     1:"one",
3     2:"two",
4     3:"three",
5 }
6 print(1 in nums)
7 print("three" in nums)
8 print(4 not in nums)

```

Listing 22: key-is-in-dict.py

```

1 #list comprehension
2 cubes=[i**3 for i in range(5)]
3
4 print(cubes)

```

Listing 23: list-comprehensions.py

```

1 sen=['i','her']
2 index=1
3 sen.insert(index,'love')
4 print(sen)

```

Listing 24: list-func-append.py

```

1 list=[1,2,3,4,5,6,2]
2 print(max(list))
3 print(min(list))
4 print(list.count(2))
5 print(list.reverse())

```

Listing 25: list-func-index.py

```

1 list=[1,2,3,4,5,6]
2 print(4 in list)
3 print(list.index(3))
4 print(list.index(7))

```

Listing 26: list-func-insert.py

```

1 numbers=list(range(1,9,2))
2 print(numbers)
3
4 print(range(20)==range(0,20))

```

Listing 27: list-func-max-min-count-rev.py

```

1 str= "Hello world!"
2 print(str)
3 print(str[4])

```

Listing 28: list-in-strings.py

```

1 nums=[1,2,3]
2 print(nums+[4, 5, 6])
3 print(nums*3)

```

Listing 29: list-reassignments.py

```

1 str= "Hello world!"
2 print(str)
3 print(str[4])

```

Listing 30: lists-.py

```

1 try :
2     word="spam"
3     print(word/0)
4 except:
5     print("An error occurred")

```

Listing 31: many-exception-handling.py

```

1 #if a function returns nothing it means the varibale stores 'None'
2 def some_func():
3     print("H!i")
4
5 var=some_func()
6 print(var)

```

Listing 32: none-in-func.py

```

1 #by default read mode
2 myfile=open("testfile.txt")
3 #read mode
4 open("testfile.txt","r")
5 #write mode
6 open("testfile.txt","w")
7 #binary wrtie mode
8 open("filename.txt","wb")
9
10 #every file which is opened must be closed
11 myfile.close()

```

Listing 33: opening-and-closing-a-file.py

```

1 msg="Hello world!"
2 file=open("newfile.txt","w")
3 amount_written=file.write(msg)
4 print(amount_written)
5 file.close()

```

Listing 34: opening-file-in-writemode-will-delete-its-content.py

Listing 35: print-no-of-length-of-strings-in-a-file.py

```
1 print"python is fun"  
2 print("python is fun!")  
3 print'Always look on the bright side'  
4 print'He\'s a very naughty boy\nI know it aunty'
```

Listing 36: print-strings.py

```
1 file=open("filename.txt","r")  
2  
3 for line in file:  
4     print(line)  
5  
6 file.close()
```

Listing 37: printing-line-by-line-in-for-loop.py

```
1 try:  
2     num = 5/0  
3 except:  
4     print("An error occured")  
5     raise
```

Listing 38: raise-exception.py

```
1 file=open("testfile.txt","r")  
2 print(file.read(16))  
3 print(file.read(4))  
4 print(file.read())  
5 file.close()
```

Listing 39: read-and-display-contents-of-the-file.py

```
1 file=open("filename.txt","r")  
2 print(file.readlines())  
3 file.close()
```

Listing 40: reading-files.py

```
1 file=open("filename.txt","r")  
2  
3 for line in file:  
4     print(line)  
5  
6 file.close()
```

Listing 41: reading-files-line-by-line.py

```
1 file=open("testfile.txt","r")  
2 print(file.read(16))  
3 print(file.read(4))  
4 print(file.read())  
5 file.close()
```

Listing 42: reading-only-certain-portion-of-a-file.py

```

1 while True:
2     print("Options:")
3     print("Enter 'add' to add two numbers")
4     print("Enter 'sub' to subtract two numbers")
5     print("Enter 'mul' to multiply two numbers")
6     print("Enter 'div' to divide two numbers")
7     print("Enter 'qte' to quit the programe")
8
9
10    if user_input=="qte":
11        break
12    elif user_input=="add":
13        num1=float(input("Enter a number: "))
14        num2=float(input("Enter another number: "))
15        result=str(num1+num2)
16        print("The answer is "+result)
17    elif user_input=="sub":
18        num1=float(input("Enter a number: "))
19        num2=float(input("Enter another number: "))
20        result=str(num1-num2)
21        print("The answer is "+result)
22    elif user_input=="mul":
23        num1=float(input("Enter a number: "))
24        num2=float(input("Enter another number: "))
25        result=str(num1*num2)
26        print("The answer is "+result)
27    elif user_input=="div":
28        num1=float(input("Enter a number: "))
29        num2=float(input("Enter another number: "))
30        result=str(num1/num2)
31        print("The answer is "+result)
32    else:
33        print("Unkown input")

```

Listing 43: sim-calc.py

```

1 print("Janani "+" loves "+"Elango")

```

Listing 44: simpe-int.py

```

1 while True:
2     print("Options:")
3     print("Enter 'add' to add two numbers")
4     print("Enter 'sub' to subtract two numbers")
5     print("Enter 'mul' to multiply two numbers")
6     print("Enter 'div' to divide two numbers")
7     print("Enter 'qte' to quit the programe")
8     user_input=input(" ")
9
10    if user_input=="qte":
11        break
12    elif user_input=="add":
13        num1=float(input("Enter a number: "))
14        num2=float(input("Enter another number: "))
15        result=str(num1+num2)
16        print("The answer is "+result)
17    elif user_input=="sub":
18        num1=float(input("Enter a number: "))

```

```

19     num2=float(input("Enter another number: "))
20     result=str(num1-num2)
21     print("The answer is "+result)
22     elif user_input=="mul":
23         num1=float(input("Enter a number: "))
24         num2=float(input("Enter another number: "))
25         result=str(num1*num2)
26         print("The answer is "+result)
27     elif user_input=="div":
28         num1=float(input("Enter a number: "))
29         num2=float(input("Enter another number: "))
30         result=str(num1/num2)
31         print("The answer is "+result)
32     else:
33         print("Unkown input")

```

Listing 45: simple-calculator.py

```

1 num=12
2 if num != 12 or 5<num:
3     print("Yes")
4 else:
5     print("No")

```

Listing 46: simple-if.py

```

1 # string formatting
2 nums=[4,5,6]
3 msg="Numbers are {0} {0} {1} {2}".format(nums[0], nums[1], nums[2])
4 print(msg)
5
6 a="{x},{y}".format(x=5,y=9)
7 print(a)

```

Listing 47: string-formatting.py

```

1 print("Spam " *3)
2 print('4'*3)
3 print('python is fun ' *2)

```

Listing 48: string-operation.py

```

1 evens=[i**2 for i in range(10) if i**2 % 2==0]
2 print(evens)
3 i=range(10)
4 a=i**2
5 b=a%2
6 print(a)
7 print(b)

```

Listing 49: temp.py

```

1 filename=input("Enter a filename:")
2
3 with open(filename) as f:
4     text=f.read()
5

```



```
6 print(text)
```

Listing 50: text-analyser.py

```
1 try :
2     f=open("filename.txt")
3     print(f.read())
4 finally :
5     f.close()
```

Listing 51: try-finally-while-working-with-files.py

```
1 try :
2     word="spam"
3     print(word/0)
4 except :
5     print("An error occurred")
```

Listing 52: unknow-exception-handling.py

```
1 i=1
2 while i<=5:
3     print(i)
4     i=i+1
5 print("Finished!")
```

Listing 53: while-loop.py

```
1 i=0
2 while True:
3     i=i+1
4     if i==2:
5         print("Skipping 2")
6         continue
7     if i==5:
8         print("Breaking")
9         break
10    print(i)
11
12 print("Finished!")
```

Listing 54: while-with-break.py

```
1 with open("filename.txt") as f:
2     print(f.read())
```

Listing 55: working-with-files.py

```
1 with open("filename.txt") as f:
2     print(f.read())
```

Listing 56: working-with-files-with-statement.py

```
1 file=open("newfile.txt","r")
2 print("Reading initial contents")
3 print(file.read())
4 print("Finished")
```

```

5
6 file=open("newfile.txt","w")
7 file.write("Some new text")
8 file.close()
9
10 file=open("newfile.txt","r")
11 print(file.read())
12 print("Finished")
13 file.close()

```

Listing 57: writting-in-a-file.py

```

1 %this writes the table in notes.md file automatically reading log
  file%
2 system('bash rename.sh')
3 copyfile('src.tex','master.tex')
4 f_notes='master.tex';
5
6 %copyfile(f_notes,'lammps_log');
7 %cd lammps_log
8 py_list=dir('*.py');
9 m_list=dir('*.m');
10 n=length(py_list);
11 m=length(m_list);
12
13 % —— first insert emty lines —— %
14 for i=1:n
15     str1='\lstinputlisting[language=Python, caption=%caption%]{%
      filename%}';
16     ins_str(f_notes,str1)
17     str_caption=strrep(py_list(i).name,'_','-');
18     flag2file(f_notes,'%caption%',str_caption)
19     flag2file(f_notes,'%filename%',py_list(i).name)
20     %ins_str(f_notes,'\clearpage')
21 end
22 for i=1:m
23     str1='\lstinputlisting[language=Octave, caption=%filename%]{%
      filename%}';
24     ins_str(f_notes,str1)
25     flag2file(f_notes,'%filename%',m_list(i).name)
26     %ins_str(f_notes,'\clearpage')
27 end
28 % —— end ——%
29
30 ins_str(f_notes,'\lstlistoflistings')
31
32 ins_str(f_notes,'\end{document}')
33
34 %%
35 %% —— sub func —— %%
36
37 function ins_str(filename,str)
38     fid=fopen(filename,'a');
39     fprintf(fid,'\n%s',str);
40     fclose(fid);
41
42 end
43 % —— sub func —— %

```

```

44
45 % reading a file into cell array
46 function flag2file(f_name, flag, v_name)
47 fid=fopen(f_name, 'r');
48 nn=num2str(v_name);
49 rep_times=1;
50 i=1;
51 tline = fgetl(fid);
52 A{i} = tline; replace=rep_times;
53 while ischar(tline)
54     if logical(strfind(A{i}, flag)) & replace <= rep_times
55         A{i}=strrep(A{i}, flag, nn);
56         replace=replace+1;
57     end
58     i=i+1;
59     tline=fgetl(fid);
60
61     A{i} = tline;
62 end
63 fclose(fid);
64
65 % writing a file into file from cell array
66 fid = fopen(f_name, 'w');
67 for i=1:numel(A)
68     if A{i+1} == -1
69         fprintf(fid, '%s', A{i});
70         break
71     end
72     fprintf(fid, '%s\n', A{i});
73 end
74 end

```

Listing 58: makeflash.m

## Listings

1	1-print.py . . . . .	1
2	1-print-quo-rem.py . . . . .	1
3	agumending-list-and-repeat-list.py . . . . .	1
4	concatenation.py . . . . .	1
5	def-func.py . . . . .	1
6	dictioaries.py . . . . .	1
7	dictionaries-adding-element.py . . . . .	1
8	exception-andling.py . . . . .	2
9	file-rereading-will-return-nothing.py . . . . .	2
10	finally.py . . . . .	2
11	float-to-int-type-conversion.py . . . . .	2
12	fucn-euqals-variable.py . . . . .	2
13	func-as-func-input.py . . . . .	3
14	func-to-a-func.py . . . . .	3
15	func-with-inp-arg.py . . . . .	3

16	get-in-dict.py . . . . .	3
17	hello-world.py . . . . .	3
18	import-module.py . . . . .	3
19	import-more-than-two-module.py . . . . .	3
20	imprt-in-diff-name.py . . . . .	4
21	input-print.py . . . . .	4
22	key-is-in-dict.py . . . . .	4
23	list-comprehensions.py . . . . .	4
24	list-func-append.py . . . . .	4
25	list-func-index.py . . . . .	4
26	list-func-insert.py . . . . .	4
27	list-func-max-min-count-rev.py . . . . .	4
28	list-in-strings.py . . . . .	5
29	list-reassignments.py . . . . .	5
30	lists-.py . . . . .	5
31	many-exception-handling.py . . . . .	5
32	none-in-func.py . . . . .	5
33	opening-and-closing-a-file.py . . . . .	5
34	opening-file-in-writemode-will-delete-its-content.py . . . . .	5
35	print-no-of-length-of-strings-in-a-file.py . . . . .	6
36	print-strings.py . . . . .	6
37	printing-line-by-line-in-for-loop.py . . . . .	6
38	raise-exception.py . . . . .	6
39	read-and-display-contents-of-the-file.py . . . . .	6
40	reading-files.py . . . . .	6
41	reading-files-line-by-line.py . . . . .	6
42	reading-only-certain-portion-of-a-file.py . . . . .	6
43	sim-calc.py . . . . .	7
44	simpe-int.py . . . . .	7
45	simple-calculator.py . . . . .	7
46	simple-if.py . . . . .	8
47	string-formating.py . . . . .	8
48	string-operation.py . . . . .	8
49	temp.py . . . . .	8
50	text-analyser.py . . . . .	8
51	try-finally-while-working-with-files.py . . . . .	9
52	unknow-exception-handling.py . . . . .	9
53	while-loop.py . . . . .	9
54	while-with-break.py . . . . .	9
55	working-with-files.py . . . . .	9
56	working-with-files-with-statement.py . . . . .	9
57	writting-in-a-file.py . . . . .	9
58	makeflash.m . . . . .	10