CODES:

January 13, 2019

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

Listing 1: 1-print.py

```
quotient=20//3;
reminder=16%5;
print(quotient);
print(reminder);
print(7%(5//2));
```

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

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

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

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

Listing 4: concatenation.py

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

my_func()
```

Listing 5: def-func.py

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

Listing 6: dictioaries.py

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

Listing 7: dictionaries-adding-element.py

```
try:
    num1=7
    num2=7
    print(num1/num2)
    print("Done Calculation!")
except ZeroDivisionError:
    print("An error occurred")
    print("due to zero division")
```

Listing 8: exception-andling.py

```
file=open("testfile.txt","r")
print("Reading")
print(file.read())
print("Re-reading")
print(file.read())
```

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

```
try:
    print("Hello")
    print(1/0)
except ZeroDivisionError:
    print("Dived by zero")
finally:
    print("This code will run no matter what")
```

Listing 10: finally.py

```
print("2"+"3")
a=int("2")+int(3)
b=4+7
c=int(3.1)+int(5.8)
d="3"*2
print(a)
print(b)
print(c)
print(d)
e=int(d)
print(e)
```

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

```
def add(x,y):
    return x+y

def do_twice(func,x,y):
 return func(func(x,y),func(x,y))

a=5
b=6

print(do_twice(add,a,b))
```

Listing 12: fucn-euqals-variable.py

```python
def shout(word):
    """
    Print a word with an exclamation mark following itself.
    """
    print(word+"!")

shout("ellai")
```

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

```python
import random

for i in range(5):
    value=random.randint(10,16)
    print(value)
```

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

```python
def even(x):
    if x%2==0:
        print("yes")
    else:
        print("No")

even(9)
even(6)
```

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

```python
pairs={
1:"apple",
"orange":[2, 3, 4],
True:False,
None:"True",
}

print(pairs.get("orange"))
print(pairs.get(7))
print(pairs.get(1,"ae its there"))  #this results in false
print(pairs.get(12345,"this is not in dict"))
```

Listing 16: get-in-dict.py

```python
print(10/50)
```

Listing 17: hello-world.py

```python
from math import pi,sqrt

print(pi)
print(sqrt(5))
```

Listing 18: import-module.py

```python
from math import sqrt as square_root
print(square_root(3))
```

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

```python
from math import sqrt as square_root
print(square_root(3))
```

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

```python
name=input("enter something:")
print("Nice to meet you"+name)
```

Listing 21: input-print.py

```python
nums={
1:"one",
2:"two",
3:"three",
}
print(1 in nums)
print("three" in nums)
print(4 not in nums)
```

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

```python
#list comprehension
cubes=[i**3 for i in range(5)]

print(cubes)
```

Listing 23: list-comprehensions.py

```python
sen=['i','her']
index=1
sen.insert(index,'love')
print(sen)
```

Listing 24: list-func-append.py

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

Listing 25: list-func-index.py

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

Listing 26: list-func-insert.py

```python
numbers=list(range(1,9,2))
print(numbers)

print(range(20)==range(0,20))
```

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

```
str= "Hello world!"
print(str)
print(str[4])
```

Listing 28: list-in-strings.py

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

Listing 29: list-reasignments.py

```
str= "Hello world!"
print(str)
print(str[4])
```

Listing 30: lists-.py

```
try:
    word="spam"
    print(word/0)
except:
    print("An error occurred")
```

Listing 31: many-exception-handling.py

```
#if a function returns nothing it means the varibale stores 'None'
def some_func():
    print("H!i")

var=some_func()
print(var)
```

Listing 32: none-in-func.py

```
#by default read mode
myfile=open("testfile.txt")
#read mode
open("testfile.txt","r")
#write mode
open("testfile.txt","w")
#binary wrtie mode
open("filename.txt","wb")

#every file which is opened must be closed
myfile.close()
```

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

```
msg="Hello world!"
file=open("newfile.txt","w")
amount_written=file.write(msg)
print(amount_written)
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

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

Listing 36: print-strings.py

```
file=open("filename.txt","r")

for line in file:
    print(line)

file.close()
```

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

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

Listing 38: raise-exception.py

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

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

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

Listing 40: reading-files.py

```
file=open("filename.txt","r")

for line in file:
    print(line)

file.close()
```

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

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

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

```
while True:
    print("Options:")
    print("Enter 'add' to add two numbers")
    print("Enter 'sub' to subtract two numbers")
    print("Enter 'mul' to multiply two numbers")
    print("Enter 'div' to divide two numbers")
    print("Enter 'qte' to quit the programe")


    if user_input=="qte":
        break
    elif user_input=="add":
        num1=float(input("Enter a number: "))
        num2=float(input("Enter another number: "))
        result=str(num1+num2)
        print("The answer is "+result)
    elif user_input=="sub":
     num1=float(input("Enter a number: "))
     num2=float(input("Enter another number: "))
     result=str(num1-num2)
     print("The answer is "+result)
    elif user_input=="mul":
     num1=float(input("Enter a number: "))
     num2=float(input("Enter another number: "))
     result=str(num1*num2)
     print("The answer is "+result)
    elif user_input=="div":
     num1=float(input("Enter a number: "))
     num2=float(input("Enter another number: "))
     result=str(num1/num2)
     print("The answer is "+result)
    else:
        print("Unkown input")
```

Listing 43: sim-calc.py

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

Listing 44: simpe-int.py

```
while True:
    print("Options:")
    print("Enter 'add' to add two numbers")
    print("Enter 'sub' to subtract two numbers")
    print("Enter 'mul' to multiply two numbers")
    print("Enter 'div' to divide two numbers")
    print("Enter 'qte' to quit the programe")
    user_input=input(" ")

    if user_input=="qte":
        break
    elif user_input=="add":
        num1=float(input("Enter a number: "))
        num2=float(input("Enter another number: "))
        result=str(num1+num2)
        print("The answer is "+result)
    elif user_input=="sub":
     num1=float(input("Enter a number: "))
```

```
    num2=float(input("Enter another number: "))
    result=str(num1-num2)
    print("The answer is "+result)
  elif user_input=="mul":
    num1=float(input("Enter a number: "))
    num2=float(input("Enter another number: "))
    result=str(num1*num2)
    print("The answer is "+result)
  elif user_input=="div":
    num1=float(input("Enter a number: "))
    num2=float(input("Enter another number: "))
    result=str(num1/num2)
    print("The answer is "+result)
  else:
      print("Unkown input")
```

Listing 45: simple-calculator.py

```
num=12
if num != 12 or 5<num:
    print("Yes")
else:
    print("No")
```

Listing 46: simple-if.py

```
# string formatting
nums=[4,5,6]
msg="Numbers are {0} {0} {1} {2}".format(nums[0], nums[1], nums[2])
print(msg)

a="{x},{y}".format(x=5,y=9)
print(a)
```

Listing 47: string-formating.py

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

Listing 48: string-operation.py

```
evens=[i**2 for i in range(10) if i**2 % 2==0]
print(evens)
i=range(10)
a=i**2
b=a%2
print(a)
print(b)
```

Listing 49: temp.py

```
filename=input("Enter a filename:")

with open(filename) as f:
    text=f.read()

```

```
print(text)
```

Listing 50: text-analyser.py

```
try:
    f=open("filename.txt")
    print(f.read())
finally:
    f.close()
```

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

```
try:
    word="spam"
    print(word/0)
except:
    print("An error occurred")
```

Listing 52: unknow-exception-handling.py

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

Listing 53: while-loop.py

```
i=0
while True:
    i=i+1
    if i==2:
        print("Skipping 2")
        continue
    if i==5:
        print("Breaking")
        break
    print(i)

print("Finished!")
```

Listing 54: while-with-break.py

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

Listing 55: working-with-files.py

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

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

```
file=open("newfile.txt","r")
print("Reading initial contents")
print(file.read())
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-reasignments.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