K. J. Somaiya College of Engineering, Mumbai-77

Department of Computer Engineering

Roll No.: 1911027

Experiment No. 4

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with

Problem Statement:
Implement the Lexical analyzer for C language and Construct Symbol Table, Function Table, Operator Table, Keyword Table.

AIM: Use concepts of python file handling

Expected OUTCOME of Experiment:

CO1: Describe the Numbers, Math functions, Strings, List, Tuples and Dictionaries in Python.

CO2: Interpret different Decision Making statements, Functions, Object oriented programming in Python.

CO3: Understand and summarize different File handling operations.

Books/ Journals/ Websites referred:

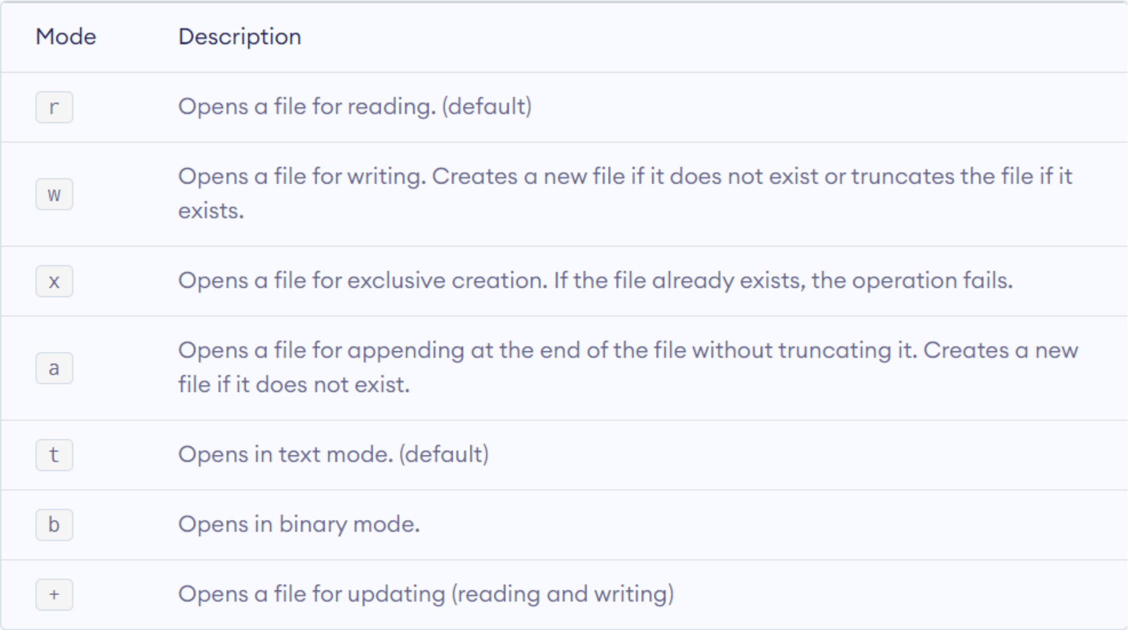
1) https://www.geeksforgeeks.org/file-handling-python/
2) https://www.w3schools.com/python/python_file_handling.asp
3) https://www.programiz.com/python-programming/file-operation
4) https://www.geeksforgeeks.org/open-a-file-in-python/

Pre Lab/ Prior Concepts:

File Handling:

Python too supports file handling and allows users to handle files i.e., to read and write files, along with many other file handling options, to operate on files. The concept of file handling has stretched over various other languages, but the implementation is either complicated or lengthy, but alike other concepts of Python, this concept here is also easy and short. Python treats file differently as text or binary and this is important. Each line of code includes a sequence of characters and they form text file. Each line of a file is terminated with a special character, called the EOL or End of Line characters like comma {,} or newline character. It ends the current line and tells the interpreter a new one has begun. File handling is an important part of any web application. Python has several functions for creating, reading, updating, and deleting files.

There are different methods (modes) for opening a file:



Mode	Description
r	Opens a file for reading. (default)
w	Opens a file for writing. Creates a new file if it does not exist or truncates the file if it exists.
x	Opens a file for exclusive creation. If the file already exists, the operation fails.
a	Opens a file for appending at the end of the file without truncating it. Creates a new file if it does not exist.
t	Opens in text mode. (default)
b	Opens in binary mode.
+	Opens a file for updating (reading and writing)

Opening a file: Opening a file refers to getting the file ready either for reading or for writing. This can be done using the open() function. This function returns a file object and takes two arguments, one that accepts the file name and another that accepts the mode(Access Mode).

Syntax: File_object = open(‘File_Name’, ‘Access_Mode’)

This will create a file object with the access mode specified. Any operation on the file can be done with this File_object now.

For e.g: f=open(‘MyFile.txt’)

This will create a handle to the file ‘MyFile.txt’. We can perform any operation on file using this handle now.

Reading from a file: This can be done by following access methods:

1) Read Only (‘r’) : Open text file for reading. The handle is positioned at the beginning of the file. If the file does not exists, raises I/O error. This is also the default mode in which file is opened.

2) Read and Write (‘r+’) : Open the file for reading and writing. The handle is positioned at the beginning of the file. Raises I/O error if the file does not exists.

There are three ways to read data from a text file:

1) read(): Returns the read bytes in form of a string. Reads n bytes, if no n specified, reads the entire file.

Syntax: File_object.read([n])

2) readline(): Reads a line of the file and returns in form of a string. For specified n, reads at most n bytes. However, does not reads more than one line, even if n exceeds the length of the line.

Syntax: File_object.readline([n])

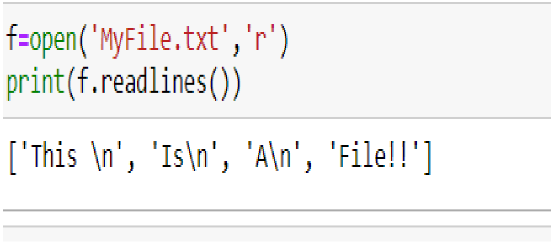
3) readlines(): Reads all the lines and return them as each line a string element in a list.

Syntax: File_object.readlines()

For e.g: f=open('MyFile.txt','r')

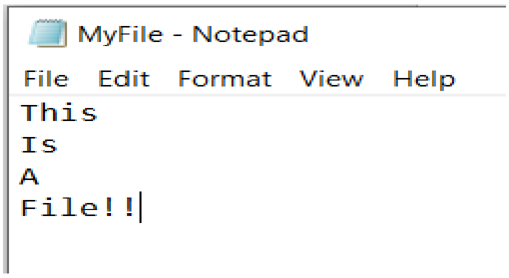
Print(f.readlines())

This will print a list representation of all the lines in the file 'MyFile.txt'.



```
f=open('MyFile.txt','r')
print(f.readlines())
```

```
['This \n', 'Is\n', 'A\n', 'File!!']
```



```
This
Is
A
File!!
```

Writing to a file: This can be done by following access methods:

1) write only ('w'): Open the file for writing. For existing file, the data is truncated and over-written. The handle is positioned at the beginning of the file. Creates the file if the file does not exists.

2) write and read ('w+'): Open the file for reading and writing. For existing file, data is truncated and over-written. The handle is positioned at the beginning of the file.

There are 2 ways to write data to the file:

1) write(): Inserts the string str1 in a single line in the text file.

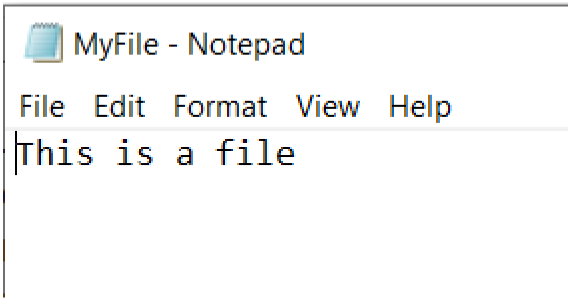
Syntax: File_object.write(str)

2) writelines(): For a list of string elements, each string is inserted in the text file. Used to insert multiple strings at a single time.

Syntax: File_object.writelines(L) for L = [str1, str2, str3]

For e.g: f=open('MyFile.txt','w')

f.write("This is a file")



```
This is a file
```

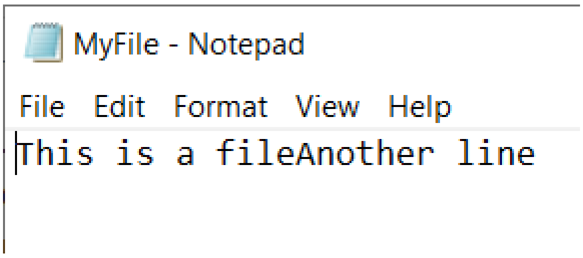
Appending to a file: This can be done by following access methods:

1) append only ('a'): Open the file for writing. The file is created if it does not exist. The handle is positioned at the end of the file. The data being written will be inserted at the end, after the existing data.

2) append and read ('a+'): Open the file for reading and writing. The file is created if it does not exist. The handle is positioned at the end of the file. The data being written will be inserted at the end, after the existing data.

For e.g: f=open('MyFile.txt','a')

f.write("Another line")

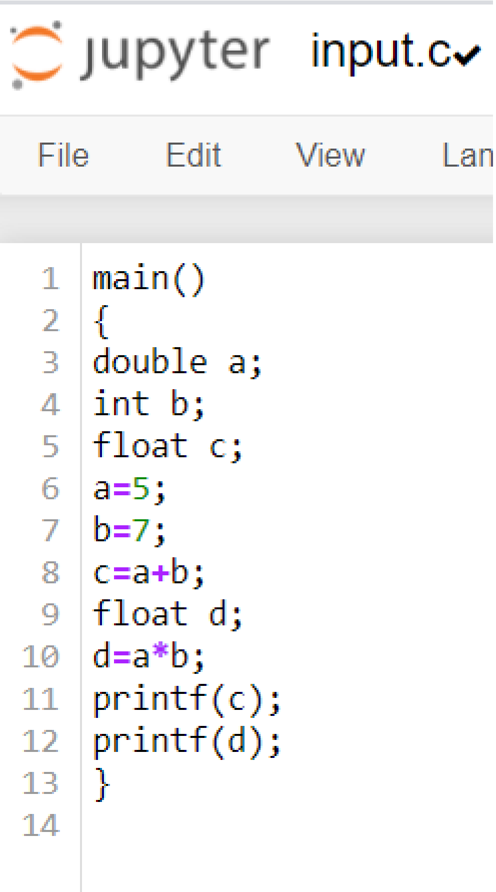


```
This is a fileAnother line
```

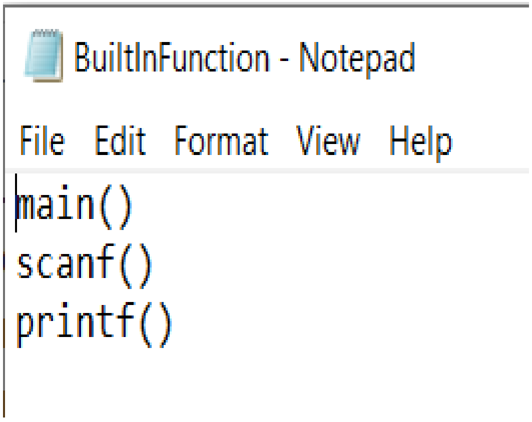
Closing a file: close() function closes the file and frees the memory space acquired by that file. It is used at the time when the file is no longer needed or if it is to be opened in a different file mode.

Syntax: File_Object.close()

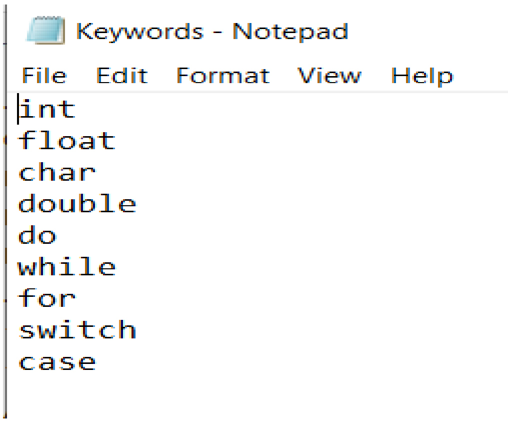
Requirements:



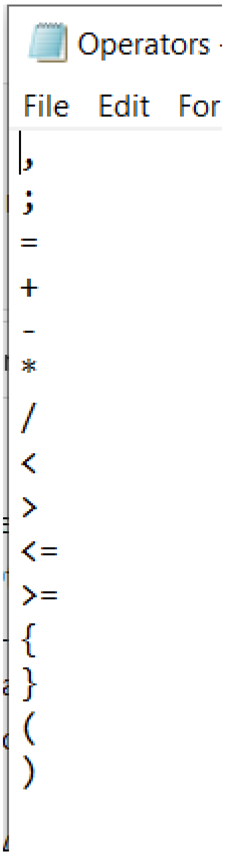
```
main()
{
double a;
int b;
float c;
a=5;
b=7;
c=a+b;
float d;
d=a*b;
printf(c);
printf(d);
}
```



```
main()
scanf()
printf()
```



```
int
float
char
double
do
while
for
switch
case
```



```
,
;
=
+
-
*
/
<
>
<=
>=
{
}
(
)
```

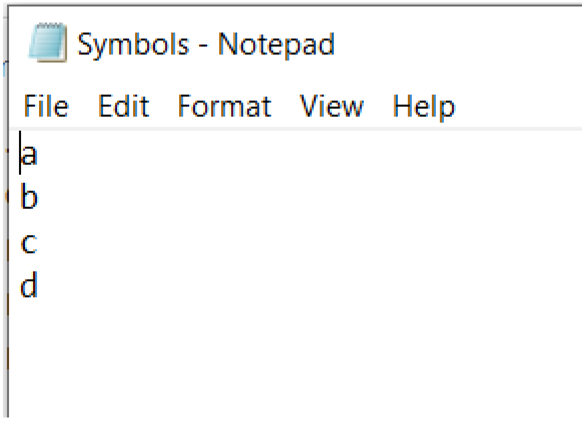
Program:

```
def check(temp):
    for i in range(0,len(temp)):
        for t in oper:
            if(temp[i]==t[:-1]):
                return True
    return False


def bui_check(temp):
    for i in range(0,len(buif)):
        if(temp==buif[i][:-3]):
            return True,(i+1)
    return False,0

def key_check(temp):
    for i in range(0,len(keyw)):
        if(temp==keyw[i][:-1]):
            return True,"K",(i+1)
    return False,"X",0

def ope_check(temp):
    for i in range(0,len(oper)):
        if(temp==oper[i][:-1]):
            return True,"O",(i+1)
    return False,"X",0

def sym_check(temp):
    f6=open("Symbols.txt",'r')
    ls=f6.readlines()
    f6.close()
    for i in range(0,len(ls)):
        if(temp==ls[i][:-1]):
            return True,(i+1)
    return False,0

def con_check(temp):
    f7=open("Constants.txt",'r')
    ls=f7.readlines()
    f7.close()
    for i in range(0,len(ls)):
        if(temp==ls[i][:-1]):
            return True,(i+1)
    return False,0

f=open('input.c','r')
f1=open('BuiltInFunction.txt','r')
f2=open('Keywords.txt','r')
f3=open('Operators.txt','r')
f5=open('Constants','w')
inp=f.readlines()
buif=f1.readlines()
keyw=f2.readlines()
oper=f3.readlines()
for line in inp:
    lst=line.split()
    for temp in lst:
        flag=0
        flag1=0
        flag2=0
        a,b=bui_check(temp[:6])
        if(a==True):
            print("Func#",b,end=" ")
            flag1=1
        a,b=bui_check(temp[:5])
        if(a==True):
            print("Func#",b,end=" ")
            flag2=1
        a,b=bui_check(temp[:4])
        if(a==True):
            print("Func#",b,end=" ")
            continue
        if(flag1==1 or flag2==1):
            if(flag1==1):
                cap=""
                for i in temp[6:]:
                    cap=cap+i
            else:
                cap=""
                for i in temp[5:]:
                    cap=cap+i
            temp=cap
        c=check(temp)
        if(c==True):
            l=list()
            for i in range(0,len(temp)):
                l.append(temp[i])
            for i in l:
```

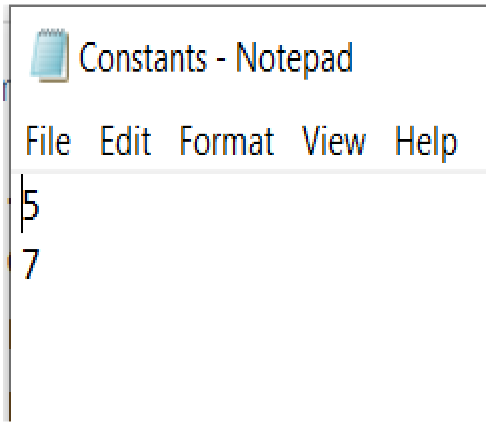
```
flag=0
a,b,c=key_check(i)
if(b=="X"):
    a,b,c=ope_check(i)
    if(b!="X"):
        print("Ope#",c,end=" ")
        flag=1
else:
    print("Key#",c,end=" ")
if(flag==0):
    if(i.isdigit()==False):
        str=i+"\n"
        a,b=sym_check(i)
        if(a==True):
            print("Sym#",b,end=" ")
        else:
            f4=open('Symbols.txt','a')
            f4.write(str)
            f4.close()
            a,b=sym_check(i)
            print("Sym#",b,end=" ")
    else:
        str=i+"\n"
        a,b=con_check(i)
        if(a==True):
```

```
                            print("Con#",b,end=" ")
                        else:
                            f7=open('Constants.txt','a')
                            f7.write(str)
                            f7.close()
                            a,b=con_check(i)
                            print("Con#",b,end=" ")
                else:
                    a,b=bui_check(temp[:-2])
                    if(a==True):
                        print("Func#",b,end=" ")
                    a,b,c=key_check(temp)
                    if(a==True):
                        print("Key#",c,end=" ")
    print("\n-------------------------------------------------------------------------------------------------------")
f.close()
f1.close()
f2.close()
f3.close()
f5.close()
```

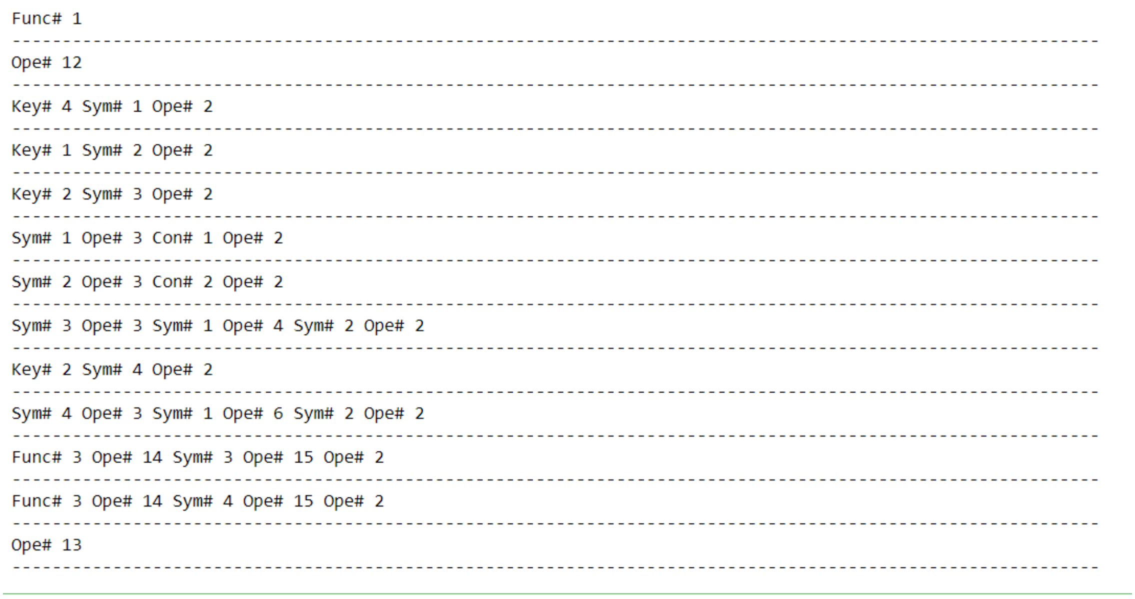
Output:



```
a
b
c
d
```



```
5
7
```



```
Func# 1
------------------------------------------------------------------------------------------------------------
Ope# 12
------------------------------------------------------------------------------------------------------------
Key# 4 Sym# 1 Ope# 2
------------------------------------------------------------------------------------------------------------
Key# 1 Sym# 2 Ope# 2
------------------------------------------------------------------------------------------------------------
Key# 2 Sym# 3 Ope# 2
------------------------------------------------------------------------------------------------------------
Sym# 1 Ope# 3 Con# 1 Ope# 2
------------------------------------------------------------------------------------------------------------
Sym# 2 Ope# 3 Con# 2 Ope# 2
------------------------------------------------------------------------------------------------------------
Sym# 3 Ope# 3 Sym# 1 Ope# 4 Sym# 2 Ope# 2
------------------------------------------------------------------------------------------------------------
Key# 2 Sym# 4 Ope# 2
------------------------------------------------------------------------------------------------------------
Sym# 4 Ope# 3 Sym# 1 Ope# 6 Sym# 2 Ope# 2
------------------------------------------------------------------------------------------------------------
Func# 3 Ope# 14 Sym# 3 Ope# 15 Ope# 2
------------------------------------------------------------------------------------------------------------
Func# 3 Ope# 14 Sym# 4 Ope# 15 Ope# 2
------------------------------------------------------------------------------------------------------------
Ope# 13
------------------------------------------------------------------------------------------------------------
```

Conclusion: By performing this experiment we successfully understood concepts of file handling(opening a file, reading a file, writing to a file and appending to a file) and functions in python. Also implemented a program of lexical analyser of C using python and its file handling ability.

Date: 23 / 3 / 2021 Signature of faculty in-charge

Post Lab Descriptive Questions

Q-1. Which of the following statements correctly explain the function of tell() method?

A. tells the current position within the file.
B. indicates that the next read or write will occur at that many bytes from the beginning of the file.
C. move the current file position to a different location.
D. it changes the file position only if allowed to do so else returns an error.

ANS) A. tells the current position within the file.
B. indicates that the next read or write will occur at that many bytes from the beginning of the file.

Q-2. Which of the following statements correctly explain the function of seek() method?

A. tell the current position within the file.
B. indicate that the next read or write occurs from that position in a file.
C. determine if you can move the file position or not.
D. move the current file position to a different location at a defined offset.

ANS) D. move the current file position to a different location at a defined offset.

Q-3. Which of the following are the attributes related to a file object?

A. closed
B. mode
C. name
D. rename

ANS) A. closed
B. mode
C. name

Q-4. What will be the output of the following code snippet?

```
fo = open("myfile.txt", "w+")
print ("Name of the file: ", fo.name)
# Assuming that the file contains these lines
# TechBeamers
# Hello Viewers!!
seq="TechBeamers\nHello Viewers!!"
fo.writelines(seq )
fo.seek(0,0)
for line in fo:
    print (line)

fo.close()
```

A. TechBeamers
Hello viewers!!
B. Name of the file: myfile.txt
TechBeamers
Hello Viewers!!
C. TechBeamers Hello viewers!!
D. Syntax Error

ANS) B. Name of the file: myfile.txt
TechBeamers
Hello Viewers!!

Q-5. Which of the following statements correctly explain the function of truncate() method?

A. truncates the file’s size
B. deletes the content of the file
C. truncates the file’s size and returns that content
D. None of the above

ANS) A. truncates the file’s size

Q-6. Which of the following statements correctly define pickling in python?

A. It is a process to convert a Python object into a byte stream.
B. It is a process to convert a byte stream to Python object.
C. It is done using two methods dump and load.
D. Serialization is an alternate name for pickling.

**ANS) A. It is a process to convert a Python object into a byte stream.
C. It is done using two methods dump and load.
D. Serialization is an alternate name for pickling.**

Q-7. What is the correct syntax of open() function?

A. file = open(file_name [, access_mode][, buffering])
B. file object = open(file_name [, access_mode][, buffering])
C. file object = open(file_name)
D. None of the above

ANS) B. file object = open(file_name [, access_mode][, buffering])

Q-8. What will be the output of the following code snippet?

```
# Open a file in read-write mode
fo = open("myfile.txt", "w+")
print ("Name of the file: ", fo.name)

# Assuming file has the following line
txt = "This is 1st line,"
fo.writelines( txt )
```

```
seq = " This is 2nd line, This is 3rd line"
# Write sequence of lines at the end of the file.
fo.seek(0, 2)
fo.writelines( seq )

# Now read complete file from beginning.
fo.seek(0,0)
line = fo.readlines()
print ("Read Line: %s" % (line))

# Close the file
fo.close()
```

A. Name of the file: myfile.txt
Read Line: [‘This is 1st line, This is 2nd line, This is 3rd line’]

B. Name of the file: myfile.txt
Read Line: [‘ This is 2nd line, This is 3rd line’]

C. Read Line: [‘This is 1st line’]
D. Runtime Error

ANS) A. Name of the file: myfile.txt
Read Line: [‘This is 1st line, This is 2nd line, This is 3rd line’]

Q-9. What will be the output of the following code snippet?

```
with open("hello.txt", "w") as f:
    f.write("Hello World how are you today")

with open('hello.txt', 'r') as f:
    data = f.readlines()
    for line in data:
        words = line.split()
        print (words)
    f.close()
```

A. Runtime Error
B. Hello World how are you today
C. [‘Hello’, ‘World’, ‘how’, ‘are’, ‘you’, ‘today’]
D. Hello

ANS) C. [‘Hello’, ‘World’, ‘how’, ‘are’, ‘you’, ‘today’]

Q-10. What will be the output of the following code snippet?

```
f = None

for i in range (5):
    with open("myfile.txt", "w") as f:
        if i > 2:
            break

print (f.closed)
```

A. Runtime Error
B. True
C. False
D. Hello world

ANS) B. True

Q-11. What will be the output of the following code snippet?

```
f = open("data.txt", "r")
txt = "This is 1st line\n"
f.writelines( txt )
f.seek(0,0)
line = f.readlines()
print ("Read Line: %s" % (line))
f.close()
```

A. [‘ This is 1st line\n’]
B. []
C. IO Error
D. None

ANS) C. IO Error

Q-12. What will be the output of the following code snippet?

```
colors = ['red\n', 'yellow\n', 'blue\n']
f = open('colors.txt', 'w')
f.writelines(colors)
f.close()
f.seek(0,0)
for line in f:
    print (line)
```

A. red
yellow
blue
B. [‘red\n’, ‘yellow\n’, ‘blue\n’]
C. Error: I/O operation on closed file.
D. Compilation error

ANS) C. Error: I/O operation on closed file.