Q. P. Code 1-70055.

AS3

F.Y.B.Sc. Computer Science
Semester II Examination April 2019
Programming with Python – II : Solution set

Note: Any point other than solution set which is relevant to the question; as per evaluator's knowledge; can be subject for marks.

Q. 1 Attempt all. (Each of 5 marks) (15M)

(a) Multiple Choice Questions (Choose the correct alternative.) (5M)

(i) The _____ method is used to display the current working directory.
(a) getcwd()
(b) chdir()
(c) mkdir()
(d) rmdir()

(ii) Grid is used for _____.
(a) Positioning widgets at absolute locations.
(b) Arranging widgets.
(c) Packing widgets into a cavity.
(d) All ot the above.

(iii) Python exceptions are caught by the _____ keyword.
(a) catch
(b) except
(c) try
(d) throw

(iv) This method binds the socket to address.
(a) socket.socket()
(b) socket.accept()
(c) socket.bind(address)
(d) socket.listen()

(v) The _____ is a standard tkinter widget used to implement one of many selections buttons.
(a) Option button
(b) CheckBox Button
(c) Radio Button
(d) None of the mentioned

(b) Fill in the blanks. (5M)

{TCP/IP, UDP, SMTP, FTP, os.remove('file'), os.delete('file'), readline(), readlines(), focus, create }

(i) To draw things in the canvas, use the create methods to add new items.
(ii) readline() function is used to read single line from file.
(iii) os.remove() deletes a file.
(iv) SMTP protocol is used to send mail.
(v) Connected less communication can be establish using UDP.

(c) Short Answers Write answers in one or two lines. (5M)

(i) Finally clause
A finally clause is always executed before leaving the try statement, whether an exception has occurred or not

(ii) Attributes of file object
file.closed, file.mode, file.name, file.softspace

(iii) mysql connector
MySQL Connector/Python is a standardized database driver for Python platforms and development.

(iv) Socket() function
The socket() function returns a *socket object* whose methods implement the various socket system calls.

(2)

(v) Connection oriented communication in python
TCP/ IP is used for connection oriented communication. Socket object will be created using following function.
socket.socket(socket.AF_INET, socket.SOCK_STREAM)

Q.2 Attempt the following:(ANY THREE) (Each of 5 marks) (15M)

(a) Write a python program to read contents of 'first.txt' file and write same content in 'second.txt' file.
[5 marks for program]

```
file1=open('first.txt','r')
s=file1.read()
file2=open('second.txt','w')
file2.write(s)
file1.close()
file2.close()
```

(b) What are various modes of file object? Explain any five.
[1 mark for each file mode with explanation]

Sr.No.	Modes & Description
1	**r** Opens a file for reading only. The file pointer is placed at the beginning of the file. This is the default mode.
2	**rb** Opens a file for reading only in binary format. The file pointer is placed at the beginning of the file. This is the default mode.
3	**r+** Opens a file for both reading and writing. The file pointer placed at the beginning of the file.
4	**rb+** Opens a file for both reading and writing in binary format. The file pointer placed at the beginning of the file.
5	**w** Opens a file for writing only. Overwrites the file if the file exists. If the file does not exist, creates a new file for writing.
6	**wb** Opens a file for writing only in binary format. Overwrites the file if the file exists. If the file does not exist, creates a new file for writing.

7	**w+** Opens a file for both writing and reading. Overwrites the existing file if the file exists. If the file does not exist, creates a new file for reading and writing.
8	**wb+** Opens a file for both writing and reading in binary format. Overwrites the existing file if the file exists. If the file does not exist, creates a new file for reading and writing.
9	**a** Opens a file for appending. The file pointer is at the end of the file if the file exists. That is, the file is in the append mode. If the file does not exist, it creates a new file for writing.
10	**ab** Opens a file for appending in binary format. The file pointer is at the end of the file if the file exists. That is, the file is in the append mode. If the file does not exist, it creates a new file for writing.
11	**a+** Opens a file for both appending and reading. The file pointer is at the end of the file if the file exists. The file opens in the append mode. If the file does not exist, it creates a new file for reading and writing.
12	**ab+** Opens a file for both appending and reading in binary format. The file pointer is at the end of the file if the file exists. The file opens in the append mode. If the file does not exist, it creates a new file for reading and writing.

(c) Define Iterator with example.

[2 marks for Iterator]

[3 marks for example]

An iterator is an object that contains a countable number of values.

An iterator is an object that can be iterated upon, meaning that you can traverse through all the values.

Technically, in Python, an iterator is an object which implements the iterator protocol, which consist of the methods __iter__() and __next__().

For example

```
mytuple = ("apple", "banana", "cherry")
myit = iter(mytuple)
print(next(myit))
print(next(myit))
print(next(myit))
```

(d) Explain exception handling with example using try, except, raise keywords.
[2 marks for explanation]
[3 marks for example]
Exception Handling – This would be covered in this tutorial. Here is a list standard Exceptions available in Python
An exception is an event, which occurs during the execution of a program that disrupts the normal flow of the program's instructions. In general, when a Python script encounters a situation that it cannot cope with, it raises an exception. An exception is a Python object that represents an error.
When a Python script raises an exception, it must either handle the exception immediately otherwise it terminates and quits.
Handling an exception
If you have some *suspicious* code that may raise an exception, you can defend your program by placing the suspicious code in a try: block. After the try: block, include except: statement, followed by a block of code which handles the problem as elegantly as possible.

(e) Give syntax and example of following functions.
(i) mkdir
(ii) chdir
[2 marks for mkdir]
[3 marks for chdir]
We can make a new directory using the mkdir() method.
This method takes in the path of the new directory. If the full path is not specified, the new directory is created in the current working directory.
We can change the current working directory using the chdir() method.
The new path that we want to change to must be supplied as a string to this method. We can use either forward slash (/) or the backward slash (\) to separate path elements.
It is safer to use escape sequence when using the backward slash.

(f) Differentiate between match() and search() function. Explain with example.
[2 marks for explanation]
[3 marks for example]
The re.match() method finds match if it occurs at start of the string.
The re.search() method is similar to re.match() but it doesn't limit us to find matches at the beginning of the string only.

Q.3 Attempt the following:(ANY THREE) (Each of 5 marks)

(a) What is GUI programming? Write advantages and disadvantages. **(15M)**
[1 mark for GUI programming explanation]
[2 marks for advantages (any two)]
[2 marks for disadvantages (any two)]
Advantages of graphical user interface:-
Easiness for non-technical people:-
For non-technical people or for beginners good GUI's tends to make easiness in life. For example with few clicks on buttons user can easily make his work done. Software in shops for calculation of products sold and inventory can be better managed by an even non-technical guy. Similarly listening songs in the car is easy

for everyone.

Drag and drop feature:-

In most of the software, we have drag and drop functionality by which complex tasks are managed easily. Like dragging and dropping folders. And in mobile games, it is also nice to use. In much graphical software, drag and drop are awesome.

Looks nicer than text interface:-

In text interface, we have limited options to choose from and navigation is difficult. For non-educated people, text interface is difficult to understand and use. In GUI user can use any tool by detecting symbols or buttons.

Hotkeys usage:-

Sometimes we want a couple of functionality performed by single click then we use hotkeys. Like we see some buttons or mouse clicks/movements by which a couple of actions performed. This is very handy for speeding up tasks.

User-friendly:-

A user can easily navigate to the system without knowing a lot of details. Easy setup and ready to start working are awesome. Most of the software hides the complexity of actions from the users and display only required information is key to good interface.

Disabled people:-

In modern science, we can detect eyes movement and finger movement which is helpful for disabled people. Now, most of the software use this functionality to make life easier for disabled people. They can use software and websites easily with a few actions

Disadvantages of graphical user interface:-

Difficult to develop and high cost:-

Nice looking designs are difficult to make and may also cost extra hardware support. Like high-quality games consume a lot of device space and memory and it/ also required very skilled people to develop.

Slower than command line tools:-

In command line tools like MS dos, we perform some commands which do the work quickly. But if we do the same task in GUI then it takes extra time to complete the task.

Extra attention required:-

If we are driving a car then controlling music/radio in the car requires attention which makes our driving disturbed.

Using flat screen:-

Some graphical things do not display accurately on flat screens. In airplanes, sticks are used to control most of the things because flat screen display is not very handy. This makes the limitation of GUI.

Time consumption:-

It takes a lot of time to develop and design a good looking interface. If some bad interface builds then it makes difficult for the user to understand and use.

Memory resources:-

I see a lot of good GUI's consuming lot of memory resources which make system/device slow to perform.

Implementation:-

(6)

Testing and implementation take a lot of time. Like we may require extra software for running GUI's

(b) Write a python program to accept a number from user on entry box. Display whether entered number is even or odd on message box as user clicks on show button. **[5 marks for program]**

```
from tkinter import *
top=Tk()
def Display():
    t=int(e1.get())
    if t%2==0:
        messagebox.showinfo('Message','even')
    else:
        messagebox.showinfo('Message','odd')
e1=Entry(top,bd=5)
e1.pack()
b1=Button(top,text='show', command=Display)
b1.pack()
```

(c) Explain any five options of pack() method.
[1 mark for each option with explanation]
expand
Fill
Padx
Pady
Ipadx
Ipady
Side

(d) Discuss event handling formats. (any five)
[1 mark for each option with explanation]
<Button>
<Motion>
<ButtonRelease>
<Double-Button>
<Enter>
<Leave>
<FocusIn>
<FocusOut>
<Return>
<Key>

(e) List the widgets which provided by tkinter module. Explain any four.
[1 mark for listing]
[4 mark for explanation of any four widgets]
Button, Canvas, CheckButton, ComboBox, Entry, Frame, Label, Listbox, Menu, Message, RadioButton, Messagebox, spinbox

(f) Write a python program to display any five graphical shapes on Canvas.
[5 marks for program]

```
from tkinter import *
```

(8)

```
master = Tk()
w = Canvas(master, width=200, height=100)
w.pack()
w.create_oval(10,10,50,100,fill="red",width=1,dash=(1,1))
w.create_polygon(10,10,50,10,50,50,10,50,fill="red",outline="black",width=5)
w.create_rectangle(50, 25, 150, 75, fill="red")
w.create_arc(10,10,210,210, start=220, extent=100, style="arc",fill="red")
create_line(0, 0, 200, 100,fill="red")
mainloop()
```

Q.4 **Attempt the following:(ANY THREE) (Each of 5 marks)**

(a) Write a python program to create connection oriented server program. **(15M)**

[5 marks for program]

```
import socket
sock = socket.socket()
host = socket.gethostname()
port = 54321
sock.bind((host,port))
sock.listen(1)
clientsocket,add = sock.accept()
clientsocket.send(b'I am server')
print(clientsocket.recv(1024))
clientsocket.close()
```

(b) How to read properties of URL? Explain it with example.

[5 marks for program]

```
from urllib.request import urlopen
html = urlopen("http://www.google.com/")
d=html.info()
key=d.keys()
for i in d.items():
  print(i[0],':',i[1],'\n')
print('\n\nURL Fetched:')
print('\n',html.geturl())
```

(c) What is database connectivity in Python? Explain.

[5 marks for explanation]

The Python standard for database interfaces is the Python DB-API. Most Python database interfaces adhere to this standard.

You can choose the right database for your application. Python Database API supports a wide range of database servers such as –

- GadFly
- mSQL
- MySQL
- PostgreSQL
- Microsoft SQL Server 2000
- Informix
- Interbase

- Oracle
- Sybase

The DB API provides a minimal standard for working with databases using Python structures and syntax wherever possible. This API includes the following –

- Importing the API module.
- Acquiring a connection with the database.
- Issuing SQL statements and stored procedures.
- Closing the connection

We have to interact to database in two ways:

1. Type commands into a GUI, you type commands into a python interpreter.
2. Write programs in python. These programs import a library and use that library to create tables, insert records and retrieve the data you want.

(d) Give a short note on SMTP. Explain how to create SMTP object in python.
[3 marks for short note on SMTP]
[2 marks for SMTP object creation]

Simple Mail Transfer Protocol (SMTP) is an Internet standard for email transmission. SMTP was first defined in 1982 by RFC 821, and updated in 2008 by RFC 5321 to Extended SMTP additions; which is the protocol in widespread use today. Mail servers and other mail transfer agents use SMTP to send and receive mail messages on TCP port 25. Proprietary systems such as Microsoft Exchange and IBM Notes and webmail systems such as Outlook.com, Gmail and Yahoo! Mail may use their own non-standard protocols internally, but all use SMTP when sending to or receiving email from outside their own systems.

```
smtpobj=smtplib.SMTP('smtp.gmail.com',587)
smtpobj.starttls()
smtpobj.login('aagharpure@gmail.com','Gopal@0510')
smtpobj.sendmail(sender,reciever,message)
```

(e) Explain exception in database connectivity in python.
[5 marks for explanation]

DataError , IntegrityError, InternalError, NotSupportedError, Operational Error Programming Error

(f) Establish a connection between python and database and write a python to display all records from Employee table(Empno, Empname, Dept, Salary). Assume records are already inserted in table.
[5 marks for program]

```
import sqlite3
conn=sqlite3.connect('emp.db')
c=conn.execute('Select * From Employee')
for i in c:
        print(i)
conn.close()
```

Q.5 Attempt the following:(ANY THREE) (Each of 5 marks) **(15M)**

(a) Explain transaction with rollback and commit in python.
[5 marks for explanation]

(9)

Python MySQL Connector provides the following method to manage database transactions.

- commit – MySQLConnection.commit() method sends a COMMIT statement to the MySQL server, committing the current transaction.
- rollback – MySQLConnection.rollback revert the changes made by the current transaction.
- AutoCommit – value can be assigned as True or False to enable or disable the auto-commit feature of MySQL. By default its value is False.

(b) What is regular expression? State and explain regular expression patterns. (any four) **[1 mark for regular expression] [4 marks for any four patterns with explanation]**

Sr. No	Pattern & Description
1	^ Matches beginning of line.
2	$ Matches end of line.
3	. Matches any single character except newline. Using m option allows it to match newline as well.
4	[...] Matches any single character in brackets.
5	[^...] Matches any single character not in brackets
6	re* Matches 0 or more occurrences of preceding expression.
7	re+ Matches 1 or more occurrence of preceding expression.
8	re? Matches 0 or 1 occurrence of preceding expression.
9	re{ n} Matches exactly n number of occurrences of preceding expression.
10	re{ n,} Matches n or more occurrences of preceding expression.
11	re{ n, m} Matches at least n and at most m occurrences of preceding expression.
12	a| b Matches either a or b.
13	(re) Groups regular expressions and remembers matched text.

14	(?imx) Temporarily toggles on i, m, or x options within a regular expression. If in parentheses, only that area is affected.
15	(?-imx) Temporarily toggles off i, m, or x options within a regular expression. If in parentheses, only that area is affected.
16	(?: re) Groups regular expressions without remembering matched text.
17	(?imx: re) Temporarily toggles on i, m, or x options within parentheses.
18	(?-imx: re) Temporarily toggles off i, m, or x options within parentheses.
19	(?#...) Comment.
20	(?= re) Specifies position using a pattern. Doesn't have a range.
21	(?! re) Specifies position using pattern negation. Doesn't have a range.
22	(?> re) Matches independent pattern without backtracking.
23	\w Matches word characters.
24	\W Matches nonword characters.
25	\s Matches whitespace. Equivalent to [\t\n\r\f].
26	\S Matches nonwhitespace.
27	\d Matches digits. Equivalent to [0-9].
28	\D Matches nondigits.
29	\A Matches beginning of string.
30	\Z Matches end of string. If a newline exists, it matches just before new line.
31	\z Matches end of string.
32	\G Matches point where last match finished.

33	\b Matches word boundaries when outside brackets. Matches backspace (0x08) when inside brackets.
34	\B Matches nonword boundaries.
35	\n, \t, etc. Matches newlines, carriage returns, tabs, etc.
36	\1...\9 Matches nth grouped subexpression.
37	\10 Matches nth grouped subexpression if it matched already. Otherwise refers to the octal representation of a character code.

(c) How to use fonts and colours in python? Specify example.

[5 marks for explanation]

Fonts are usually specified using the font widget option.

Tkinter supports a number of different font descriptor types:

1. Font descriptor: platform independent, tuple used to specify font Ex. ("Arial",10,""bold")
2. User defined font names: tkFont module provides font class
   Ex: tkFont.Font(family="Ärial", size=10, weight=tkFont.BOLD)
3. System fonts : supports system specific font names

```
from tkinter import *
top=Tk()
l1=Label(text="First", bg="Pink",font=("Comic Sans MS",10,"italic"))
l1.pack()
l2=Label(text="Second", bg="Blue",font=("Arial",12,"bold"))
l2.pack()
l3=Label(text="Third", bg="Yellow",font=("MS Sans Serif",14,"bold italic"))
l3.pack()
l4=Label(text="Forth", bg="Green",font=("Times",16,"underline"))
l4.pack()
l5=Label(text="Fifth", bg="Red",font=("Fixedsys",18,"overstrike"))
l5.pack()
l3=Label(text="Sixth", bg="White",font=("Verdana",20))
l3.pack()
top.mainloop()
```

(d) Which functions are used to find file position? Explain with example.

[1 mark for naming the functions]

[4 marks for functions with example]

The method **tell()** returns the current position of the file read/write pointer within the file.

Following is the syntax for **tell()** method – fileObject.tell()

Parameters **NA**

Return Value: **This method returns the current position of the file read/write pointer within the file.**

The method seek() sets the file's current position at the offset. The whence argument is optional and defaults to 0, which means absolute file positioning, other values are 1 which means seek relative to the current position and 2 means seek relative to the file's end.
There is no return value. Note that if the file is opened for appending using either 'a' or 'a+', any seek() operations will be undone at the next write.
If the file is only opened for writing in append mode using 'a', this method is essentially a no-op, but it remains useful for files opened in append mode with reading enabled (mode 'a+').
If the file is opened in text mode using 't', only offsets returned by tell() are legal. Use of other offsets causes undefined behavior.
Note that not all file objects are seekable.

Syntax

Following is the syntax for **seek()** method – fileObject.seek(offset[, whence])

Parameters

- **offset** – This is the position of the read/write pointer within the file.
- **whence** – This is optional and defaults to 0 which means absolute file positioning, other values are 1 which means seek relative to the current position and 2 means seek relative to the file's end.

Return Value : **This method does not return any value.**

(e) Write a short note on Layout Manager.

[5 marks for explanation]

tkinter possess three layout managers:

- pack
- grid
- place

The three layout managers pack, grid, and place should never be mixed in the same master window! Geometry managers serve various functions. They:

- arrange widgets on the screen
- register widgets with the underlying windowing system
- manage the display of widgets on the screen

Arranging widgets on the screen includes determining the size and position of components. Widgets can provide size and alignment information to geometry managers, but the geometry managers has always the final say on the positioning and sizing.