Progress in

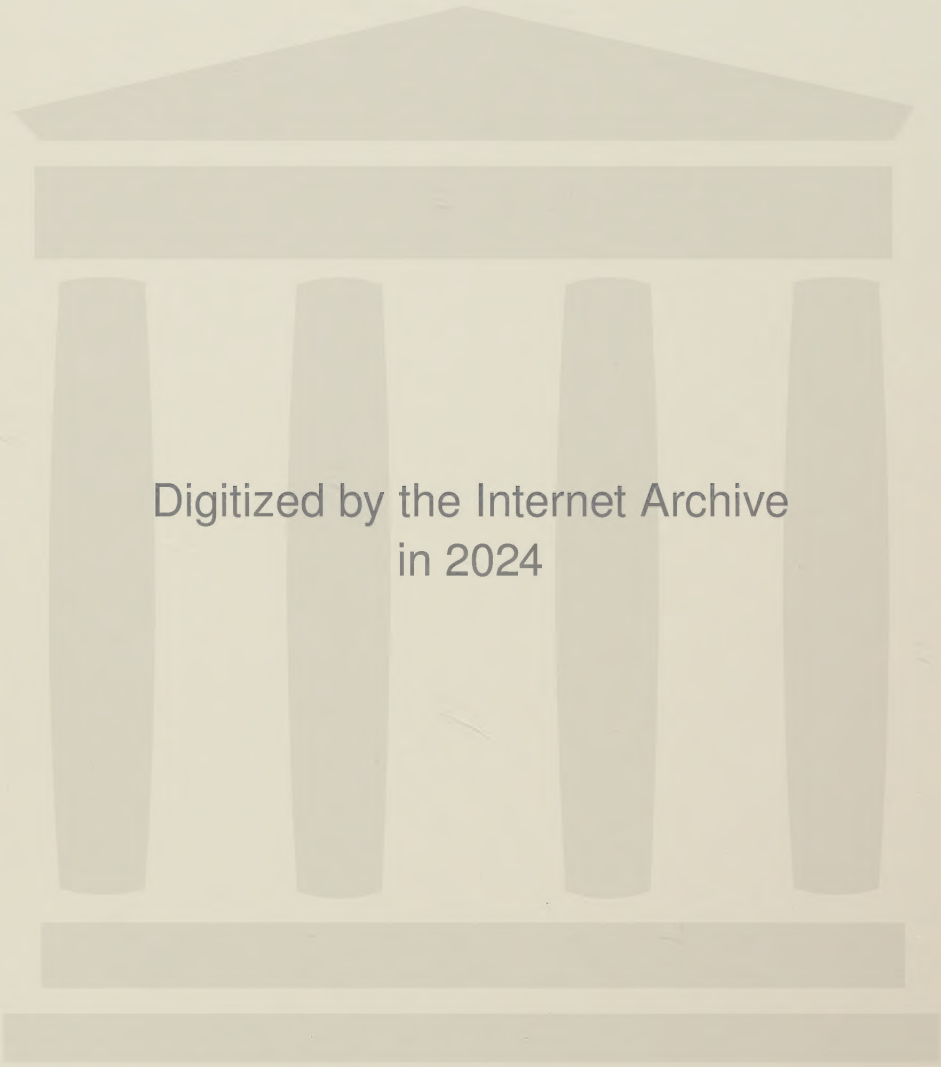
COMPUTER SCIENCE with python

Practical book for Class XII

- Computational Thinking & Programming
- Computer Networks
- Database Management

SUMITA ARORA

DHANPAT RAI & Co.



Progress in COMPUTER SCIENCE with python

[Practical Book XII]

Sumita Arora

DHANPAT RAI & CO. (Pvt.) Ltd.
EDUCATIONAL & TECHNICAL PUBLISHERS

Published by : **GAGAN KAPUR**
Dhanpat Rai & Co. (P) Ltd., Delhi

Sales Office : 1710, Nai Sarak, Delhi-110006
Phone : 2325 0251
Regd. Office : 4576/15, Agarwal Road
Darya Ganj, New Delhi-110002
Phone : 2324 7736, 37, 38, dhanpatrai@gmail.com

Disclaimer

Every effort has been made to avoid errors or omissions in this publication. In spite of this, some errors might have crept in. Any mistake, error or discrepancy noted may be brought to our notice which shall be taken care of in the next edition. It is notified that neither the publisher nor the author or seller will be responsible for any damage or loss of action to any one, of any kind, in any manner, therefrom.

EDITIONS : 1997, 1998, 2002, 2004, 2007, 2009, 2010, 2011, 2012, 2017
REPRINT : 2002, 2003, 2004, 2005, 2006, 2009, 2013, 2015, 2016,
EDITION : 2014, 2019 (*with Python*)
THIRTEENTH EDITION : 2020 (*with Python*)
REPRINT : 2021

PRICE : ₹ 595/- *Combined Price (Textbook + Practical book)*

Typesetting by : North Delhi Computer Convention, Delhi–110009,
ndcc.in@gmail.com
Printed at : Natraj Offset, Delhi

How to Use this Book ?

This is the practical component book of textbook 'Computer Science with Python' containing inclusive practical sessions that should be done during the lab-periods.

This **'Progress In Python'** practical book contains practice sessions mainly for Python unit, the ***Unit 1*** of the Textbook.

I assure you that the practice of these sessions at appropriate times as marked in the textbook would result into thorough learning and understanding of the concepts. Questions like determining outputs, find the errors, fill in blanks etc. will help you master the concepts. The practice sessions contain proper space to mark the responses during lab-practice.

Therefore, to get the full-benefit of this book along with textbook, it is advised that :

- students work on practical sessions at the right time (as suggested in the book) during lab-sessions
- fill the practice sessions space-holders ; it will help you grasp the concept more.

Important : *In some initial chapters of Python, it is advised that students solve the expressions on paper and then do them practically. It will enhance their understanding of basic essential concepts.*

Please note, this book has been designed for inclusive learning of the concepts covered in the textbook. This book cannot substitute the practical file though it would be very useful aid for it, as it contains everything your practical-file requires. So, through this, you can compile your practical-file within minutes.

So All the best !! Enjoy Python!

I would love to hear from you. You can send me your feedback, suggestions, queries, constructive criticism at my email.id as given here.

April, 2020

AUTHOR
e-mail : arora.sumita@gmail.com

Contents

Unit I : Computational Thinking and Programming

Progress in Python (Chapters 1 to 9)

Unit III : Database Management

1

Revision Tour – I

rogress in Python

PriP 1.1

Progress in Python

Python : Basic Input/Output

This PriP session is aimed at revising various concepts you learnt in Class XI.

1. *For each of the following, write a single (that is, one) Python statement* :

   (*a*) Tell the user to Enter your name, then read in, and assign to a variable, a string typed in by the user.

```
name = input("What is your name?")
```

   (*b*) Tell the user to Enter your age, then read in, and assign to a variable, an integer typed in by the user.

```
age = int(input("How old are you?"))
```

   (*c*) Print out, on separate lines, Your name is ***name*** and Your age is ***age***, where ***name*** and ***age*** are the values of the two variables entered above.

```
print("My name is", name)
print('I am', age, 'years old')
```

2. *Give three examples of legal variable names and two examples of illegal variable names in Python. For the illegal variable names explain why they are illegal. Do not use one-letter variable names.*

Legal Names
(*i*) Player_score. (*ii*) character (*iii*) player_1
Illegal Names
(*i*) 143NS → Because variable name can not start with a number
(*ii*) Nm-62 → Because we can't use special character(-) in a variable.

3. *In each of the following parts there is an error in the Python code. Identify the error by name and describe the problem. (Each piece of code is prefixed with a brief description of the programmer's intention.)*

   (*i*) The following code attempts to compute the product of **m** and **x** added to **b** and assign that value to y.

   ```
   y = mx + b
   ```

   (*b*) The following code attempts to compute the first-order effects of some physical process. You may assume that equation is correct.

   ```
   cofactor = alpha*x*x
   1storder = 1.0/cofactor
   2ndorder = 2.0 ** 1storder
   ```

4. *While taking input from input( ), you need to convert the input into float and int types, if you are reading numbers.*

   (*a*) What built-in function do you use to convert a string into a floating-point number ? Demonstrate this function by converting string '85' into a floating-point number and assigning it to a variable x.

   (*b*) What built-in function do you use to convert a string into an integer ?

   (*c*) What happens when you try to convert a string into an integer but the string is not a number ?

5. *What would be the output of following code ?*

```
print('doesn\'t')
print ("doesn\'t")
print(' "Yes," he said.')
print("\"Yes,\" he said.")
```

6. *Write a program to print an acrostic poem as shown below :*

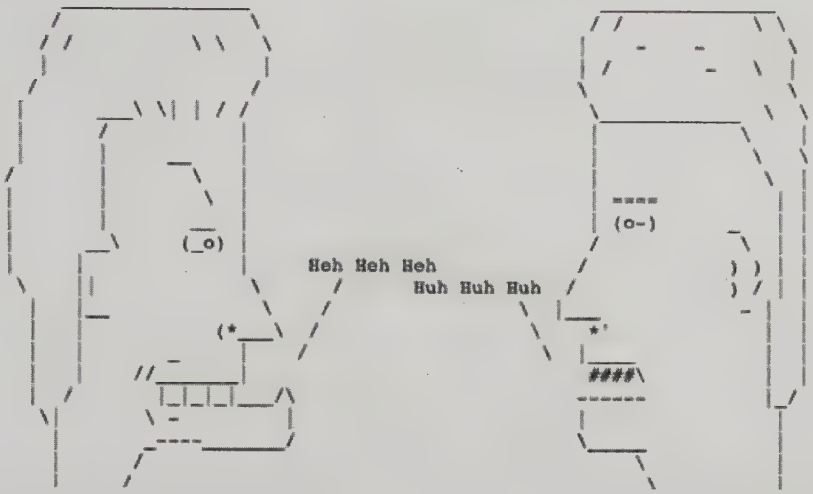
These things I have s**P**oken unto you,
that in me y**E** might have peace.
In the world ye sh**A**ll have tribulation:
but be of good **C**heer;
I have overcom**E** the world.

In the output, you need not print bigger letters for P E A C E, but make sure these letters make a vertical word in the output.

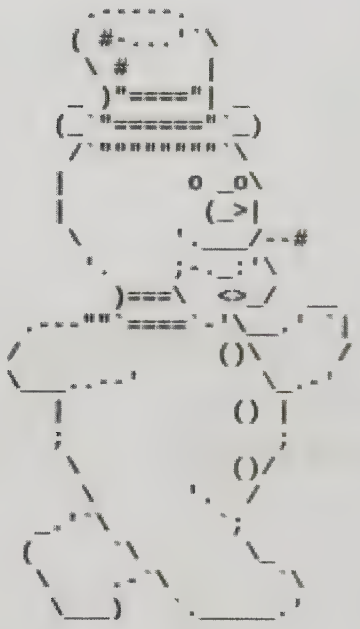
(Acrostic poems are a literary composition in which certain letters in each line form a word vertically. This acrostic poem for Peace was made using Bible verse, John 16:33.)

7. *Write a program to print following ASCII art using print statements :*

(*a*)



(*b*)



PriP 1.2

Progress in Python

Data Handling

This PriP session is aimed at revising various concepts you learnt in Class XI.

1. *In each of the following Python expressions or statements, indicate what data type belongs in the indicated place by choosing one of these data types :*

   int float bool str list

	Expression	Correct Datatype
(a)	s = int + 17	
(b)	t = float + 'pie'	
(c)	x[str] = 'catfood'	
(d)	list .sort()	
(e)	if _____ :	

2. *What does Python print as it executes the following sequence of statements ?*

	Expression	Correct Datatype
(a)	`print((10 + 3) * 2)`	26
(b)	`print(11 / 2)`	5.5
(c)	`x = 25` `print(x * 2 == 50)`	True
(d)	`print(x / 2 >= 50)`	False

(e)	`a = 'Honda'`
	`b = 'Audi'`
	`print(a + b)`
		Honda Audi

(f)	`c = len(a) + len(b)` `print(c)`	c = 5 + 4 9
(g)	`print(a)`	5
(h)	`print(a[1])`	0

3. *What would be the output of following code ?*

```
a = 9000          # initializes and assigns value to variable
print("Now it's a", type(a))
a = float(a)
print("Now it's a", type(a))
a = str(a)
print("Now it's a", type(a))
```

Integer
float
string

4. *What would be the output of following code ? Support your answer with reasons.*

```
x = 4
y = 8
z = x/y*y
print(z)
```

0.0625.

5. *Which of the following statements would yield 2? Assume that math module is imported. Recall that* ***math.fabs( )*** *gives you absolute value of its argument.*

(*a*) `print(5/2)`

(*b*) `print(5.0//2)`

(*c*) `print((int) (5.0//2))`

(*d*) `print(math.fabs( –5.0/2))`

(*e*) `print(math.fabs( –5.0//2))`

(*f*) `print(math.fabs( –3//2))`

(*g*) `print(–5/2)`

(*h*) `print(–5/2)`

(*i*) `print(–5//2)`

(*j*) `print(–5.0//2)`

6. *Which of the following statements would yield 2.0 ? Assume that* ***math*** *module is imported. Recall that* ***math.fabs( )*** *gives you absolute value of its argument.*

(*a*) `print(5/2)`

(*b*) `print(5.0//2)`

(*c*) `print((int) ( 5.0 //2))`

(*d*) `print(math.fabs( –5.0/2))`

(*e*) `print(math.fabs( –5.0//2))`

(*f*) `print(math.fabs( –3//2))`

(*g*) `print(–5/2)`

(*h*) `print(–5/2)`

(*i*) `print(–5//2)`

(*j*) `print(–5.0//2)`

7. *What would be the output produced by the print statements given in previous question ?*

8. *What would be the output of the following ?*

(*a*) `print(type([1,2]))`

(*b*) `print(type(1,2))`

(*c*) `print(type((1,2)))`

(*d*) `print(type(1/2))`

(*e*) `print(type( [1/2]))`

(*f*) `print(type( (1/2)))`

(*g*) `print(type( (1/2,)))`

(*a*)

(*b*)

(*c*)

(*d*)

(*e*)

(*f*)

(*g*)

9. *What would be the output of the following ?*

```
x = True
y = False
z = False

print(x or y and z)
```

10. *Given the following Boolean variables, what would be the output of the following statements ?*

```
x = True
y = False
z = False
```

(*a*) print(not x or y)

(*b*) print(not x or not y and z)

(*c*) print(not x or y or not y and x)

(*d*) print('ant' < 'amoeba')

(*a*)

(*b*)

(*c*)

(*d*)

11. *Write a program to find the side of a square whose perimeter you read from user.*

12. *Write a program to calculate the distance of 1056 feet in terms of yards and miles. Given fixed values are* ***1 mile = 1760 yards*** *and* ***1 yard = 3 feet***

13. *An athlete is taking rounds of a triangular park with sides as 30 m, 25 m and 35 m. The athlete has run 560 m till now. Write a program to print how many rounds of park the athlete has completed.*

14. *Write a Python program that calculates and prints on the screen the number of seconds in a year.*

PriP 1.3

Progress in Python

Control Flow Statements

This PriP session is aimed at revising various concepts you learnt in Class XI.

1. *What is the output of the following python program? First try and predict the output without the computer. Check your answer by typing it in and running it.*

```
i = 0
while i < 6:
    j = 0
    while j < i:
        print("*", end = ' ')
        j = j + 1
    i = i + 1
    print( )
```

2. *What is the output of the following program ?*

```
x = 7
y = 8
if x < 7 or x <= 10 and y > 8:
   print("IT IS!")
else:
    print("Oh No")
```

3. *What is the output of the program ?*

```
score = 40
while score > 1:
    score = score/2 - 1
    print(score, end = ' ')
```

4. *Consider the following code :*

```
if x > 3:
    if x <= 5:
        y = 1
    elif x != 6:
        y = 2
    else:
        y = 3
else:
        y = 4
```

If y has the value 2 after executing the above program fragment, then what do you know about the initial value of x ?

5. *Consider the following two fragments :*

```
if x == 5:
    x = x + 1
else:
    x = 8
```

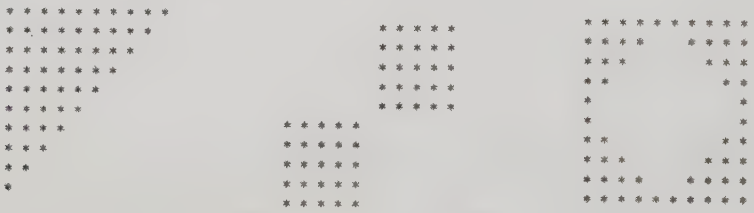
```
if x == 5:
    x = x + 1
if x != 5:
    x = 8
```

Are the two fragments logically equivalent ? Why or why not ?

6. *Write a program that prints a table on two columns: on the left the integer temperatures between 0 and 100 (Fahrenheit) and in the right column the corresponding Celsius values.*

7. *Write a program to print the following pattern :*

8. *Write programs that generate each of the patterns given below :*



(*a*) (*b*) (*c*)

(*a*)

(*b*)

(*c*)

2

Revision Tour – II

rogress in Python

PriP 2.1

Progress in Python

Python Sequences : Strings, Lists and Tuples

This PriP session is aimed at revising various concepts you learnt in Class XI.

1. *What is a string slice ? If a = "blueberry" evaluate the following :*

   (*a*) `a[2:3]` (*b*) `a[2:]` (*c*) `a[:3]` (*d*) `a[:]`

   (*e*) `a[-1:-3]` (*f*) `a[:-1]` (*g*) `a[1:1]`

   (*a*) u .

   (*b*) ueberry .

   (*c*) blu

   (*d*) blueberry .

   (*e*) _

   (*f*) blueberry

   (*g*) _

2. *Define what we mean by traversal of a string. Write a program that inputs a word (as a string, of course) and prints back the word without the vowels, like this :*

   If input word is "**nectarine**" then output should be '**nctrn**'.

   If input word is "**blueberry**" then output should be '**blbrry**'

   nectarine .

3. *Change above program to surround the vowels with parentheses instead of not showing them.*

   For input word is "**nectarine**", the output should be '**n(e)ct(a)r(i)n(e)**'

   For input word is "**blueberry**", the output should be '**bl(u)(e)b(e)rry**'

4. *Read the following pieces of code. Determine whether they will be executed successfully or not. If yes, show the result that would be printed. If not, explain why not.*

   (*a*)
   ```
   myTuple1 = (1,2,3)
   myTuple1.append(4)
   print(myTuple1)
   ```

   (*b*)
   ```
   myTuple2 = (1,2,3)
   myTuple3 = (4)
   print(myTuple2+myTuple3)
   ```

   (*c*)
   ```
   myList=[0,3,4,1]
   myList.remove(3)
   print(myList)
   ```

5. *Beside each expression, write "LEGAL" if the expression is legal python code, or "ERROR" if the expression would cause an error. Assume that myList, myTuple, and myString are a list, a tuple, and a string, respectively, and all contain at least one element.*

   1. `myList = myList + [4]`
   2. `myList.append(4)`
   3. `del myList[0]`
   4. `myTuple = myTuple + (4,)`

```
5. myTuple.append(4)
6. del myTuple[0]
7. myString = myString + "4"
8. myString.append(4)
9. del myString[0]
```

1.	6.
2.	7.
3.	8.
4.	9.
5.	

6. *Consider the following code: what is the value of a at the end ?*

```
a = [1, 2, 3]
a[2] = 0
a[a[2]] = 5
a[1:2] = []
```

7. *Write a program that accepts a list and removes the value at index 0 from the list. The program must actually modify the list passed in, and not just create a second list with the first item removed. You may assume the list you are given, will have at least one element.*

For example :

for input word is a = [1, 2, 3, 4], the output should be [2, 3, 4]

8. *Assume each of the following lines is entered in the IDLE shell one at a time. Circle any of the lines that produces an error. If no errors are found then write what is printed (from a and b).*

```
a = [[[1, 2, 3, 4, 5],[6,7,8]],9]
a[-1] = 200
b = a[:]
b[0][0][3] = 17
print(a)
print(b)
```

9. *Write a program that inputs two tuples and creates a third that contains all elements of the first followed by all elements of the second.*

   (you may use other data types such as lists etc to make your program work)

10. *Arrays in Python are implemented through lists. Write a program that sorts an array of integers in ascending order.*

11. *In context to previous program, what if, you need to sort elements of a tuple ? Is it possible ? How can you make it happen ?*

12. *Write a program that reverses a 2D array of integers, implemented as nested list, in the order of 0th element of every inner list.*

PriP 2.2

Progress in Python

Python Sequences : Dictionaries, Sorting

This PriP session is aimed at revising various concepts you learnt in Class XI.

1. *Read the following pieces of code. Determine whether they will be executed successfully or not. If yes, show the result that would be printed. If not, explain why not.*

   (*a*)
   ```
   myDictionary1={ a:1, b:2, c:3}
   myDictionary1[d] = 4
   print(myDictionary1)
   ```

   (*b*)
   ```
   myDictionary2 = { 10 : 1, 20 : 2, 30 : 3}
   print(myDictionary2[1])
   ```

2. *What would be the output of following code ?*
   ```
   mydict = {"cat":12, "dog":6, "elephant":23}
   mydict["mouse"] = mydict["cat"] + mydict["dog"]
   print mydict["mouse"]
   ```

3. *What would be the output of following code ?*
   ```
   mydict = {"cat":12, "dog":6, "elephant":23, "bear":20}
   answer = mydict.get("cat")//mydict.get("dog")
   print(answer)
   ```

4. *What would be the output of following code ?*

```
mydict = {"cat":12, "dog":6, "elephant":23, "bear":20}
print("dog" in mydict)
```

5. *What would be the output of following code ?*

```
mydict = {"cat":12, "dog":6, "elephant":23, "bear":20}
print(23 in mydict)
```

6. *What would be the output of following code ?*

```
total = 0
mydict = {"cat":12, "dog":6, "elephant":23, "bear":20}
for akey in mydict:
    if len(akey) > 3:
        total = total + mydict[akey]
print(total)
```

7. *Write code to create a Python dictionary (the dict type). Add two entries to the dictionary: Associate the key 'name' with the value 'Jivin', and associate the key 'phone' with '85000-00003'*

Add two more names and their phones in the dictionary after getting input from user. Then ask the user a name and print its corresponding phone.

8. *Write code to create a Python dictionary (the dict type) similar to previous question. Add three names and their phones in the dictionary after getting input from user. The names should act as keys and phones as their values. Then ask the user a name and print its corresponding phone.*

9. *Given a dictionary with 5 student names and their marks. Write a program that prints the dictionary contents in the ascending order of marks.*

10. Carefully read the following description and try to identify the sorting technique it is talking about.

(*a*) In this, every element is compared with its next adjacent element and if the two elements are not in proper order, the two elements are swapped in place. If there are n elements in total, then n comparisons will place the largest element at the last position. Next time, beginning with first two elements and continuing with next elements, $n-1$ comparisons will place the 2nd largest element at the 2nd last position.

(*b*) In this, we divide the list/array in two parts : (*a*) sorted part (which has no element in the beginning), (*b*) unsorted part (which has all the elements in the beginning). Every iteration is performed with an aim of putting the first element of unsorted part at the correct position in sorted part. Each such placement adds one element to sorted part and removes one element from unsorted part.

11. (*a*) Carefully go through following code of Bubble Sort technique. It is creating some problem during execution. Figure out the problem and correct it. Try writing efficient solution of the problem.

Hint. Efficient solution is error free and has less number of operation taking place

```
⋮
#arr is the list being sorted
n = len(arr)
for i in range(n):
    for j in range(n):
        if(arr[j] > arr[j + 1])
            temp = arr[j]
            arr[j] = arr[j + 1]
            arr[j + 1] = temp
for i in range(n)
        print(arr[i], end = " ")
```

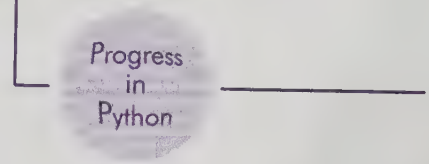
(*b*) Find out the problem in following code of Insertion sort technique. State the problem and suggest solution for it.

```
:
#arr is the list being sorted
n = len(arr)
for i in range(n):
    key = arr[i]
    j = i - 1
    while j >= 0 and key < arr[j]:
        arr[j + 1] = arr[j]
        j = j - 1
    arr[j + 1] = key
for i in range(n) :
    print(arr[i])
```

3 Working with Functions

Progress in Python

PriP 3.1



Structure of Functions and Modules

This PriP session is aimed at making anatomy of Python functions and modules clear to you.

1. *Carefully go through following sample code and identify various parts of functions.*

(*a*)
```
def sub1(a, b):
    print(b, " - ", a, " = ", b - a)
```

(*b*)
```
def cube2(x):
    x * x * x)
    return x * x * x - x
```

(*c*)
```
def cube1(x):
    return x * x * x
```

(*d*)
```
def greeting( ):
    print("Welcome to Python")
```

(*a*)		
	Function Name	
	Takes any argument? If yes, their names	
	Number of statements in the function-body	
	Returns any value?	
	What is it doing? (Carefully read the code and write a line about its functionality)	

(*b*)		
	Function Name	
	Takes any argument? If yes, their names	
	Number of statements in the function-body	
	Returns any value?	
	What is it doing? (Carefully read the code and write a line about its functionality)	

(c)	*Function Name*	
	Takes any argument? If yes, their names	
	Number of statements in the function-body	
	Returns any value?	
	What is it doing? (Carefully read the code and write a line about its functionality)	

(d)	*Function Name*	
	Takes any argument? If yes, their names	
	Number of statements in the function-body	
	Returns any value?	
	What is it doing? (Carefully read the code and write a line about its functionality)	

2. *There are some function call statements given below. Pick the function call statements for the function definitions given in question 1 :*

(a) `subtract(3, 5)`
(b) `cube2(7, x)`
(c) `sub( 5, 3)`
(d) `cube1(a, a, a)`
(e) `greetings('hello')`
(f) `sub2(8,9)`
(g) `greeting()`
(h) `sub1(x, y)`
(i) `cube1(x)`
(j) `sub2(p, q)`
(k) `sub1(5,3)`
(l) `cube2(num)`
(m) `sub1(8,9,10)`
(n) `cube(7)`

Function call statements for sub1()	
Function call statements for cube1()	
Function call statements for cube2()	
Function call statements for greeting()	

3. *If you carefully observe the function definitions of question1, you'll find functions sub1( ) and cube2( ) are basically doing the same thing – calculating difference between two values. Logically they are doing same thing but technically these are different. What is the difference ?*

(Hint : carefully look at the function bodies)

3.2

Progress in Python

Functions' Basics

This *'Progress in Python'* session is aimed at strengthening the functions' basics like: functions terminology, function anatomy, argument vs parameter and flow of execution during function calls.

1. *Identify the mentioned parts from the code given below.*

```
1. def lastDigitCube( n ) :
2.     d = n % 10
3.     c = d ** 3
4.     return c
5.
6. number = int(input( "Enter a number :" ) )
7. cube = lastDigitCube(number)
8. print("The cube of its last digit is", cube)
```

(*a*)	*Function header*	
(*b*)	*Number & name of arguments*	
(*c*)	*Number of statements in function body*	
(*d*)	*Number of statements in main program (__main__)*	
(*e*)	*Function call statement*	
(*f*)	*Parameters' name*	
(*g*)	*Arguments' name*	
(*h*)	*Flow of execution* [mention the execution order of statements *e.g.*, main.1 means main program's statement1 executed and fn.1 means function's statement1 executed. A sample order could be main.1, fn.1, main.2 etc. (This flow of execution is not for above code : p)]	

2. *Consider the following function definitions. Write appropriate function call state- ments for these.*

[**Hint :** Be careful about which function calls should be part of some expression or statement]

S.No.	Function Definition	Function Call

(*a*)	`def sumOf3Multiples1( n ) :`
	`  s = n * 1 + n * 2 + n * 3`
	`  return s`	

(*b*)	`def sumOf3Multiples2( n ) :`
	`  s = n * 1 + n * 2 + n * 3`
	`  print(s)`	

(*c*)	`def areaOfSquare( a ) :` `  return a * a`	
(*d*)	`def areaOfRectangle( a, b) :` `  return a * b`	
(*e*)	`def perimeterCircle( r ) :` `  return (2 * 3.1459 * r)`	
(*f*)	`def perimeterRectangle( l, b) :` `  return 2 * ( l + b )`	

(*g*)	`def Quote() :`
	`  print("\t Quote of the Day")`
	`  print("Act Without Expectaton!")`
	`  print("\t -Lao Tzu")`	

(*h*)	`def poly(x, y, z) :`
	`  s = x ** 3 + y ** 2 + z`
	`  return s`	

3. For the function definitions given in question 2 here, write main program segment for function definitions numbered given below :

S.No.	Function Definition	Main Program

(*a*)	`def sumOf3Multiples1( n ) :`
	`  s = n * 1 + n * 2 + n * 3`
	`  return s`	

(b)	def sumOf3Multiples2(n) :
	  s = n * 1 + n * 2 + n * 3
	  print(s)	

(c)	def Quote()
	  print("\t Quote of the Day")
	  print("Act Without Expectaton!")
	  print("\t -Lao Tzu")	

(d)	def poly(x, y, z) :
	  s = x ** 3 + y ** 2 + z
	  return s	

4. *Define a Python function called absolute that takes one parameter (x) and returns the absolute value of x, i.e., as shown below :*

   $|x| = (x \text{ if } x \geq 0, -x \text{ otherwise})$

   Don't forget to use a return statement at the end.

   Also, write the code that will demonstrate math absolute function by computing and print the absolute value of 6 and −3. Be sure to pay attention to proper indentation.

5. *Write a program with a function chkOdd( ) that takes one argument (a positive integer) and reports if the argument is odd or not.*

rip 3.3

Progress in Python

Calling Functions, Argument Types, Scope of Variables

This *'Progress in Python'* session is aimed at these concepts: *calling or invoking functions, different types of arguments, Scope of variables* and *Name resolution by Python.*

1. *Consider the following code. Identify the types and values of arguments being passed for each function call.*

```
def func( a, b = 5, c = 10 ):
    poly = 2 * a ** 2 - 5 * b + c
    print("With values", a, b, c)
    print(poly)

func( 3, 7 )                    # function call 1
func( 25, c = 24 )              # function call 2
func( c = 50, a = 100 )         # function call 3
```

Function call 1	*Types of arguments* :		
	a gets	b gets	c gets
Function call 2	*Types of arguments* :		
	a gets	b gets	c gets
Function call 3	*Types of arguments* :		
	a gets	b gets	c gets

2. *What will be the output produced by above program (given in question 1) ?*

3. *Consider following function header*

```
def robot(voltage, state = 'a stiff ', action= 'voom', type = 'Norwegian Blue'):
    :
```

For the above function, some function calls are being given below. Find the errors in these function call statements :

	Function call	State reason(s) for these calls being invalid
(*a*)	`robot()`	
(*b*)	`robot(voltage = 5.0, 'dead' )`	
(*c*)	`robot(110, voltage = 220)`	
(*d*)	`robot(actor = ' Harrison Ford ')`	

4. *What is wrong with following function code? Find reason and correct it.*

```
def addThree(n1, n2, n3):
    return n1 + n2 + n3
    print("the answer is", n1 + n2 + n3)
```

Reason	
Corrected code :	

5. *Following code is giving an error at line number 6. Why is this error occurring ? What is the solution ?*

```
1. def c_to_f(c):
2.     result = c / 5.0 * 9 + 32
3.     return result
4.
5. tempf = c_to_f(19)
6. print(result)
```

Reason	
Corrected code :	

6. *Given some programs below. Write their flow of execution.*

Program code	Flow of Execution
1. def square(x): 2. return x * x 3. 4. square(5)	
1. def square(x): 2. return x * x 3. 4. print(square(5) + square(6))	
1. def square(x): 2. return x * x 3. 4. def sum_of_squares(x, y): 5. return square(x) + square(y) 6. 7. k = sum_of_squares(2, 3) 8. print("sum is", k)	

7. *From the program given below, identify the scope of each variable in the program.*

```
x = 10                          # Reference 1
pi = 3.14

def incr(x):                    # Reference 2
    y = x + 1
    return y

def area(r):
    n = incr(r)
    ar = pi * n * n
    return ar

incr(5)
print(x)
area(4)
```

Variable Name	*Scope*

8. *Write a function that takes a character (i.e., a string of length 1) and returns* ***True*** *if it is a vowel,* ***False*** *otherwise.* (**Hint:** Make use of logical operators.)

   Make four calls to this function for testing letters 'u', 'k', 'm', 'i'.

Output (paste screenshot)

9. *A Mersenne number is a number in the form* $2^n - 1$ *i.e.,*

1st Mersenne number is $2^1 - 1 = 1$
2nd Mersenne number is $2^2 - 1 = 3$
3rd Mersenne number is $2^3 - 1 = 7$

and so on.

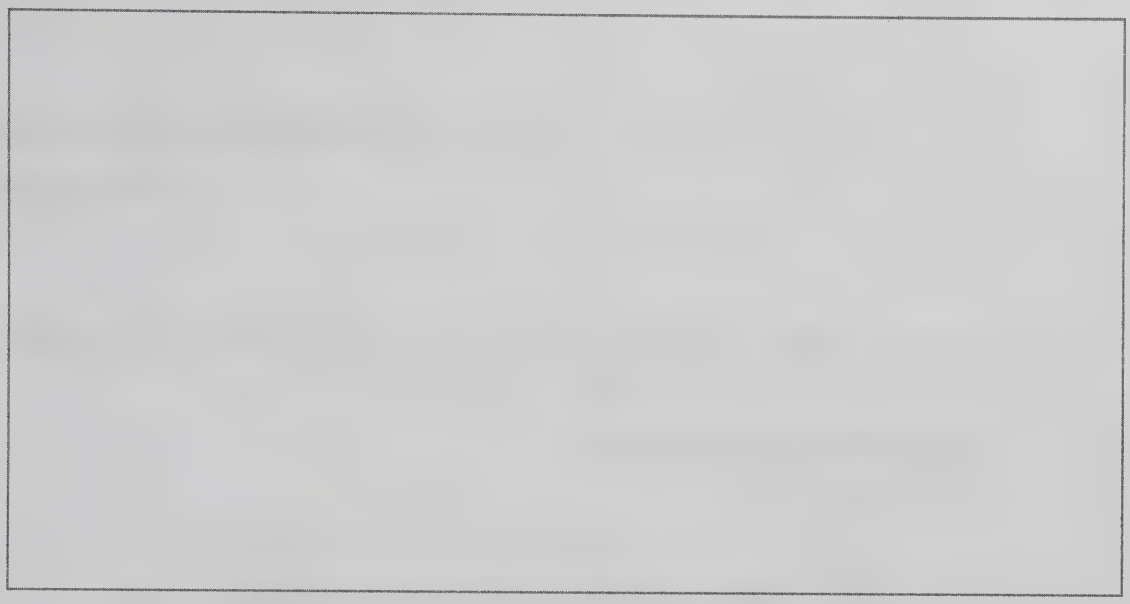
Write a program that passes value to a function mersenne( ) and the function returns nth Mersenne number.

Output (paste screenshot)

10. *Write a void function that receives a 4 digit number and calculates the sum of squares of first 2 digits' number and last two digits' number, e.g., if 1233 is passed as argument then function should calculate* $12^2 + 33^2$.

Output (paste screenshot)

11. *Write a function that takes one argument (a positive integer) and reports if the argument is prime or not. Write a program that invokes this function.*



rip 3.4

Progress in Python

Mutability/Immutability of arguments

This 'Progress in Python' session is aimed at these concepts : *Mutability/ Immutability of the arguments, parameters and their behaviour in various situations.*

1. *Consider following program code.*

```
1.  def myFunc(l, v):
2.      l[0] += 2
3.      v += 2
4.      print("Values/variables within called function :", l, v)
5.      return
6.  # __main__
7.  list1 = [1]
8.  var1 = 1
9.  print("Values/variables before function call :", list1, var1)
10. myFunc(list1, var1)
11. print("Values/variables after function call :", list1, var1)
```

(*i*) Identify flow of execution in the above program.

(*ii*) What will be the output of above program ?

(*iii*) What is reason of this output ?

2. *Consider following program code , which is similar to the previous program but the function myfunc( ) has some changes as shown below :*

```
1. def myFunc(list1, var1):
2.     list1[0] += 2
3.     var1 += 2
4.     print("Values/variables within called function :", list1, var1)
5.     return
   :
   :   (rest of the code (lines 6-11) is exactly the same)
```

(*i*) Identify flow of execution in the above program.

(*ii*) What will be the output of above program ?

(*iii*) What is reason of this output ?

(*iv*) Try making memory environment(s) for above program.

3. *Consider following program code.*

```
1. def myFunc(l, v):
2.     l[0] += 2
3.     v += 2
4.     N = [2, 3, 4]
5.     l = N
6.     l.append(7)
7.     print(l, v)
8.    return
9. # __main__
10. list1 = [1]
11. var1 = 1
12. print("Values/variables before function call :", list1, var1)
13. myFunc(list1, var1)
14. print("Values/variables after function call : ", list1, var1)
```

(*i*) Identify flow of execution in the above program

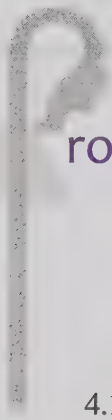
(*ii*) What will be the output of above program ?

(*iii*) What is reason of this output ?

(*iv*) Try making memory environment(s) for above program.

4

Using Python Libraries



rogress in Python

4.1

Progress in Python

Creating and Using Modules

1. *Consider the code of module* ***welcome.py*** *given below*

```
#welcome.py
"""
welcome
~~~~~
This module contains the greeting message 'msg' and greeting function 'greet()'.
"""
msg = 'Hello'     # Global Variable
def greet(name):  # Function
    print('{}, {}'.format(msg, name))
```

Now carefully go through the code statements given below and figure out what will be their output. Every successive statement should assume that previous code statements hold i.e., part (a) holds for (b); parts (a), (b) hold for (c) and so on.

(*a*)
```
>>> import welcome
```

(*b*)
```
>>> welcome.greet('Parth')
>>> print(welcome.msg)
```

(*c*)
```
>>> welcome.__doc__
>>> welcome.__name__
```

(*d*) `>>> help(welcome)`

(*e*)
```
>>> import welcome as wel
>>> wel.greet('Radha')
```

2. *Create a module* ***conversion.py*** *that should contain the following :*

Constant K = 273.15 # 0 degree C = 273.15 kelvin

Two conversion functions :

function Cels_to_Fahr(tmp)

That converts given temperature in Celsius to temperature in Fahrenheit using formula :

°F = (9 °F/5 °C)°C + 32 °F

and function Fahr_to_Cels(tmp)

That converts given temperature in Fahrenheit to temperature in Celsius using formula :

°C = (5 °C/9 °F)(°F - 32 °F)

Add proper docstring to the module.

(*a*) Code for conversion.py

(*b*) `help(conversion)` **# give its output below as per your code of conversion.py**

rip 4.2

Progress in Python

Working with math and random Modules

1. *Area of an equilateral triangle can be computed as* $\frac{\sqrt{3}}{4}a^2$ *where* **a** *is its side. Write a program to obtain side from user and print the area of equilateral triangle with side* **a**.

Python's result **(screenshot)**

2. *Area of regular polygon can be computed as :*

$$\frac{1}{2} n \sin\left(\frac{360°}{n}\right) S^2$$

where n = number of sides

S = length from centre to a corner.

Write a program to obtain values ***n*** *and* ***S*** *from user and print area of polygon.*

Python's result (**screenshot**)

3. (*a*) *In Python shell, type the following :*

```
import math
```

Now type the following statements one by one and log what happens :

(*i*) `math.sqrt(49)`

(*ii*) `sqrt(49)`

(*iii*) `math.fabs(- 3)`

(*iv*) `fabs (- 3)`

(*i*)

(*ii*)

(*iii*)

(*iv*)

(*b*) *Now quit the Python shell. Again restart Python IDLE shell and type*

```
from math import*
```

Type the same set of statements given in part (a) i.e., (i) to (iv). Log what happens

(*i*)

(*ii*)

(*iii*)

(*iv*)

4. *Write a program to randomly generate a number from the range 100 – 1000 both inclusive and print the generated number as :*

```
randomly selected lucky viewer is : <number>
```

Python's result **(screenshot)**

Prip 4.3

Progress in Python

Crating and using Packages

1. *Mark created following directory structure for his module namely chk :*

```
chk
├── work.py
└── test
    └── newwork.py
```

He has copied the folder's link to site-packages folder of Python but still the following command is giving error :

```
In [1]: import chk
Traceback (most recent call last):

  File "<ipython-input-1-c6d47c0933ec>", line 1, in <module>
    import chk

ModuleNotFoundError: No module named 'chk'
```

(*a*) What is the possible reason of above error ?

(*b*) Rectify above problem.

(*c*) After rectifying, write command to import work module of above shown chk package

(*d*) After rectifying, write command to import newwork module of above shown chk pacakge

2. *Create a package as per following :*

```
mailsys/
|--- __init__.py
|--- deliver.py  # contains function send_mail(), get_mail(), draft(), outbox(), inbox()
└─ archive
    └─ __init__.py
    └─ Oldmail.py  # contains functions fiveyr(), starred()
```

After creation, this should be importable in a program with import statement. It should also have proper docstring so that help() function gives proper information about it.

(*a*) Complete code of delivery.py (make assumptions)

(*b*) Complete code of oldmail.py (make assumptions)

(*c*) Write command to display details of module delivery

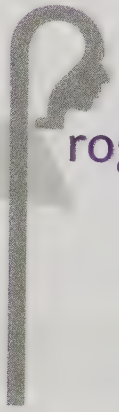
(*d*) Write command to import get_mail( ) function

(*e*) Write command to import these modules : *deliver* and *oldmail*

(*f*) Write command to run draft() function

(*g*) Write command to run fiveyr( ) function

5 File Handling



Progress in Python

5.1 Data Files in Python

5.1

Progress in Python

Data Files in Python

Consider the file "sarojiniPoem.txt"

Autumn Song

Like a joy on the heart of a sorrow,
The sunset hangs on a cloud;
A golden storm of glittering sheaves,
Of fair and frail and fluttering leaves,
The wild wind blows in a cloud.

Hark to a voice that is calling
To my heart in the voice of the wind:
My heart is weary and sad and alone,
For its dreams like the fluttering leaves have gone,
And why should I stay behind?

Sarojini Naidu

1. *Based on above file, answer the following questions :*

   (a) *What would be the output of following code :*

```
file = open("sarojiniPoem.txt","r")
text = file.readlines()
file.close()

for line in text:
   print(line, end = '')
print()
```

(*b*) *Modify the program so that the lines are printed in reverse order.*

(*c*) *Modify the code so as to output to another file instead of the screen. Let your script overwrite the output file.*

(*d*) *Change the script of part (c) so that it appends the output to an existing file.*

(*e*) *Modify the program so that each line is printed with a line number at the beginning.*

2. *Write a program to print following type of statistics for the given file :*

```
16824  lines in the file
  483  empty lines
 53.7  average characters per line
 65.9  average characters per non-empty line
```

3. *Write a program that asks a new user about userid and password and then appends it to file "security.txt" provided the given userid does not exist in the file. If it does, then display error message "User id already exists" and prompts the user to re-enter the userid. Also, make sure that the password is at least 8 characters long with atleast a digit and a special character out of "$, @, and %" in it.*

4. *Given a file "mine.txt" as shown below :*

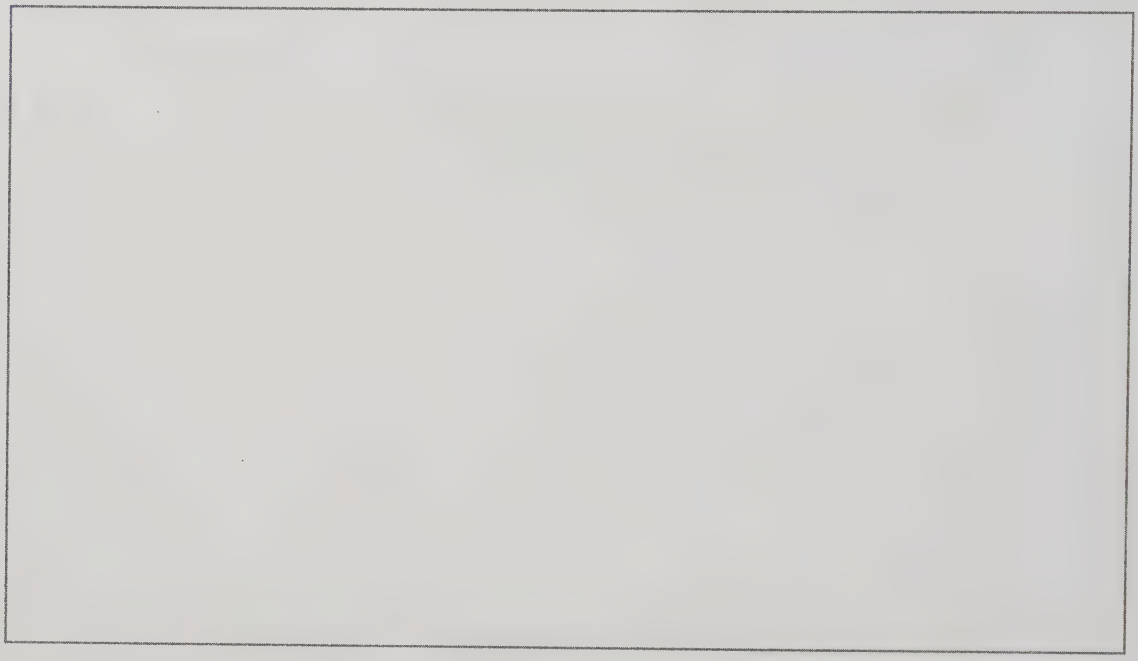
```
I am Line 1

I am Line 3
```

What would be the output of following code ?

```
print(file("mine.txt").readlines())
```

5. *Write a program that prompts the user for a file name and then reads and prints the contents of the requested file in upper case.*

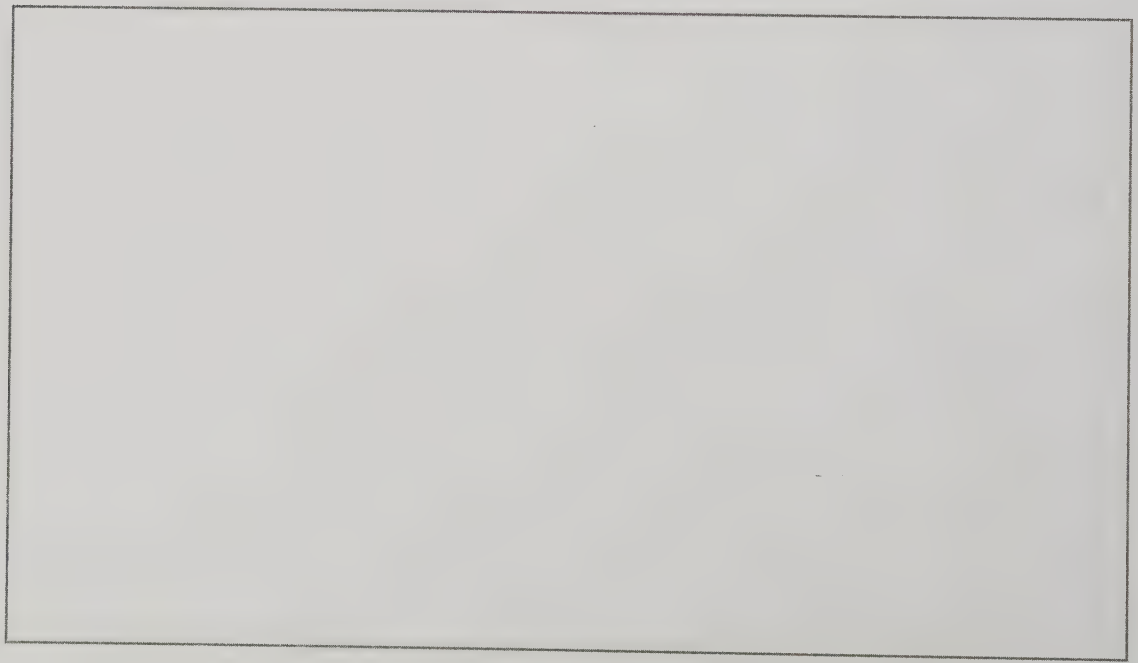
6. *Write a program that reads a text file and then creates a new file where each character's case is inverted.*



7. *Create a file phonebook.det that stores the details in following format :*

```
Name        Phone
Jivin       86666000
Kriti       1010101
:
```

Obtain the details from the user.



8. *Write a program to append more details to file "phonebook.det".*

9. *Write a program to edit the phone number of "Arvind" in file "phonebook.det". If there is no record for "Arvind", report error.*

10. *Write a Python program to create a binary file* ***Items.dat*** *to store Item details of some items as per { itemno, name, price, category }.*

11. *Write a program to append some records to the binary file Items.dat, created in the previous program.*

12. *Write a Python program to modify the price of an item in the binary file Items.dat created data. Get the itemno of the item to be modified from the user.*

13. *Write a Python program to create a binary file Itemnew.dat storing all the item details of binary file* ***Items.dat*** *except for the item whose itemno is obtained from the user, e.g., if user gives Itemno as 1005, then* ***Itemnew.dat*** *should store all 'items' details of* ***Items.dat*** *except item 1005.*

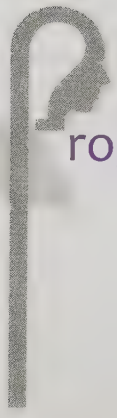
14. *Write a Python program to get 10 items' details (itemno, name, price, category) from the user and create a CSV file (Items.csv).*

15. *Write a Python program to create a csv file (Itemnew.csv) same as the previous program, but with a delimiter character as pipe (|)*

16. *Write a program to read a file* ***Itemnew.csv*** *and search for an item whose itemno is obtained from the user, as the keyboard input.*

17. *Write a program to read from a file Items.csv and create a file highitems.csv, containing only those item details from Items.csv where price > 250.*

6 Recursion



Progress in Python

6.1 Recursion Practically

6.1

Progress in Python

Recursion Practically

1. **Understanding recursion.** *Consider following function* ***num_vowels()*** *which takes a string* **s** *of lower-case letters and returns the number of vowels in the string parameter. Fill the missing functionality.*

```
def num_vowels(s):
    """s – a string of 0 or more lower-case letters"""
    if s == '' :
        return 0
    else:
        # write here recursive call to the function
        ----------------------------
        if s[0] in 'aeiou':
            return 1 + ----------------
        else:
            return 0 + num_in_rest
```

2. **Tracing Recursion.** Consider below given is a recursive function **myst( )** that takes two integer parameters. Trace the execution of the following call to this function :

```
myst(1, 32)
```

The function is :

```
def myst(a, b):
    if a >= b:
        return b
    else:
        myst_r = myst(a * 2, b // 2)
        return a + myst_r
```

Space for making and tracing call stack is given below :

- We have filled in some of the components of the first two calls for you.
- You should add sections for additional calls as needed, all the way down to the base case.
- Next, if the call is a base case, simply show the return value
- If the call is a recursive case, show the recursive call on the line for myst_r (you can refer to first filled sample).
- After executing the base case, work your way back through the sections for the previous calls. Add in both the results of the recursive call (i.e, the value assigned to myst_r and the value returned by the call itself.

```
myst(1, 32)
------------
    a = 1,    b = 32
    myst_r = myst(2, 16) = ...
    return ...
----------------------------------------
    myst (2, 16)
--------------
        a = ?,        b = ?
        myst_r = ...
        return ...
----------------------------------------
:
        myst (        )
        --------------
        a = ...
        b = ...
        myst_r = ...
        return ...
```

3. **Debugging recursion.** *Consider following recursive function that returns the number of consonants in string s passed to it as parameter. Parameter s contains a lowercase string. The function below is not working properly. Debug the code below so that it works as desired.*

```
def num_consonants(s):
    if s == ' ':
        return 0
    else:
        num_in_rest = num_consonants(s[1:])
        if s[0] not in 'aeiou':
            return num_in_rest
        else:
            return num_in_rest + 1
```

You can use these common debugging techniques (refer to your previous class' learning) :

(*i*) Spot the origin of error

(*ii*) Print variables' intermediate values

(*iii*) Trace code step by step

4. **Designing recursive code.** *A Geometric Progression (GP) is a progression where the each term is a multiple of the previous one. The multiplying factor is called the common ratio.*

   So a GP with a first term **a** *and a common ratio* **r** *with* **n** *terms, can be stated as :*

   $a, ar, ar^2, ar^3, ar^4 \ldots ar^{n-1}$

   (a) Write a recursive function that prints a GP. Input a, r and n in __main__ part.

(*b*) Write a recursive function that calculates the sum of a GP by changing the function that you wrote in part(*a*). Obtain a, r and n in __main__ part.

Highlight the changes that were made to get the desired result.

5. We can determine how many digits a positive integer has by repeatedly dividing by 10 (without keeping the remainder) until the number is less than 10, consisting of only 1 digit. We add 1 to this value for each time we divided by 10.

Implement this recursive functionality in Python and test it using a main function that calls this with the values 15, 105, and 15105.

Hint. Remember, in Python 3.x if n is an integer, n/10 will not be an integer.

6. Write a recursive Python function that has a parameter representing a list of integers and returns the maximum stored in the list. Thinking recursively, the maximum is either the first value in the list or the maximum of the rest of the list, whichever is larger. If the list only has 1 integer, then its maximum is this single value, naturally.

   Hint. If A is a list of integers, and you want to set the list B to all of the integers in A except the first one, you can write B = A[1:len(A)]

8

Data Structures – I : Linear Lists

rogress in Python

8.1 Searching, Insertion, Deletion in the List

8.1

Progress in Python

Searching, Insertion, Deletion in the List

1. List Operations. *Supposing that the command LList=['Cats', 'rule', '!', '?', '!', '?'] has been executed, complete the following tables as usual, using Python interactive mode to verify whether you were right. Remember: list slicing produces a new list object.*

S.No.	Commands	Expected Output	Were you right ?
1.	`LList.remove('!')` `print(LList)`		
2.	`LList.remove('C')`		
3.	`print(LList[4])`		
4.	`print(LList[5])`		
5.	`LList.insert(1, 'may')` `print(LList)`		
6.	`print(LList.index('?'))`		
7.	`print(LList.index('?!'))`		

8.	`copy1 = LList`
	`copy2 = LList[:]`
	`LList[0] = 'Dogs'`
	`print(copy1)`		

9.	`print(copy2)`		
10.	`copy2.append('And also drool.')` `print(copy2)`		
11.	`print(len(copy2))`		

2. *Consider the following sorted array :*

```
data = [1, 4, 6, 7, 9, 10, 14]
```

(a) *If using linear/sequential search, how many comparisons (i.e., how many elements in the array do we need to examine) are needed to determine that 9 is in the array ?*

(*b*) *If using linear/sequential search, how many comparisons (i.e., how many elements in the array do we need to examine) are needed to determine that 11 is not in the array ?*

(*a*)

(*b*)

3. *Fill in the blanks in the following code segment to implement sequential search of the given array. The method should return a boolean value indicating whether or not the parameter value was found in the array.*

```
def seqSearch( s ) :
    a = [9, 12, 14, 3, 25 ]
    for i in _____ :
        if _____ :
            return _____
    return _____
```

4. *Write a function to insert an element in a sorted list only if it does not already exist in the list.*

5. *Write a function to insert an element in a sorted list only if it already exists in the list. (i.e., only duplicate entries can be inserted)*

6. *Write a function to delete an element from a sorted list.*

7. *Modify the function deleting an element from a sorted list so that only duplicate entries can be removed. An element which is unique in the list cannot be deleted, e.g., from the array given below only 10, 35, 43 can be deleted as these have multiple entries ; other elements cannot be deleted from it*

```
[5, 10, 10, 12, 20, 35, 35, 35, 40, 42, 43, 43, 50, 70]
```

8. **A** is a two dimensional list created through following code :

```
rows = int(input("Rows:"))
cols= int(input("Columns:"))
A = [ ([0] * cols) for row in range(rows)]
A[0][0] = 42
```

Now consider following codes and determine what will be the output ?

Consider this	What output do you anticipate ?	Were you right ? (run to confirm)
`B = [row if row[0]!=0 else row*2 for row in A ]` `print (B)`		

`rows = len(A)`
`cols = len(A[0])`
`print ("rows =", rows)`
`print ("cols =", cols)`		

9. *Create a two dimensional List in Python that stores runs scored by a batsmen in five overs. The list looks somewhat like :*

```
Runs = [ [0, 6, 4, 1, 0, 0], [3, 0, 2, 0, 0, 0], [0, 0, 4, 4, 0, 1],
         [0, 0, 0, 1, 0, 0], [0, 0, 0, 0, 0, 0]]
```

(*a*) Write a function that returns the over in which the batsman scored the highest runs.

(*b*) Write a function that returns the over(s) in which the batsman scored the minimum runs.

(*c*) Write a function that returns the total runs scored by the batsman.

9

Data Structures – II : Stacks and Queues Using Lists

rogress in Python

PriP 9.1

Progress in Python

Working with Stacks

This PriP session is dedicated to the practice of data structure Stack's concepts and provides practice assignment for the same.

1. *Given a stack holding some years. Answer the following questions :*

   (*a*) Which stack element(s) can be accessed ?

   (*b*) Name three operations which can be performed on a stack.

   (*c*) A stack is called a LIFO structure. What does this mean ?

   (*d*) Give an example of an application of the stack.

2. *Show the Stack's status after each of the following operations. Show position of top by underlining the element. Stack s is initially empty.*

   (*a*) s.push(20)
   (*b*) s.push(51)
   (*c*) s.pop( )
   (*d*) s.pop( )
   (*e*) s.pop( )
   (*f*) s.push(43)

(*a*)	(*b*)
(*c*)	(*d*)
(*e*)	(*f*)

3. *Evaluate following postfix expression using a stack :*

```
30, 5, 2, **, 12, 6, /, +, - , 3
```

4. *Add a function StackSize() to above program that returns the number of elements on the stack.*

5. *Write a program to implement a stack of years (4 digit years) storing the years when Olympics were held in Asia or Europe.*

6. *Write a program to print a string in reverse order.*

   Hint : *Extract individual characters from string and push in a stack ; once done, then keep popping from stack.*

7. *A palindrome is a string which is the same whether the characters are read from left to right or from right to left. For example, "radar" , "deed", and "able was I ere I saw elba" are all examples of palindromes. There is a simple algorithm to determine whether a string is a palindrome given here :*

```
push each letter of the string on a stack.  Also place the character
into a character array, say str.
Set j = 0 and done = false
While the stack is not empty and not(done)
    pop a character, say ch
    if str[j] == ch then
        increment j and continue
    else set done to true (the string isn't a palindrome)
if done is true
    the string isn't a palindrome
else it is.
```

Write a Python program implementing the above algorithm.

PriP 9.2

Progress in Python

Working with Queues

This PriP session is dedicated to the practice of data structure Queue's concepts and provides practice assignment for the same.

1. *After the following operations have all been completed : draw the picture of the resulting queue.*

(*a*) Q.Enqueue(20)

(*b*) Q.Enqueue(51)

(*c*) Q.Enqueue(31)

(*d*) Q.Enqueue(71)

(*e*) Q.Dequeue( )

(*f*) Q.Enqueue(43)

(*g*) Q.Dequeue( )

2. *Determine whether each of the following characteristics apply to a stack (S), a queue (Q), both (B), or neither (N) :*

   (*a*) An element is inserted at a special place called the top.

   (*b*) An element is inserted at a special place called the rear.

   (*c*) An element is deleted at the front.

   (*d*) The i^{th} position may be deleted.

   (*e*) An element is deleted at the top.

   (*f*) The structure is a LIFO structure.

   (*g*) The structure is a FIFO structure.

   (*h*) The structure is a restricted access structure.

3. *Write a program to implement a queue of order numbers in a restaurant. Display a menu as shown below :*

```
YUMMY PROGRAM
  1. Order a meal
  2. Waiting Queue
  3. Order is ready
  4. Exit
```

- When option 1 is chosen, generate an order number of three digits, randomly and add it to your queue.
- For option 2, display the orders that are in queue.
- For option 3, dequeue an order number from the queue and flash it like :

```
Order number <orderno> is ready e.g.,
Order number 345 is ready.
```

4. *A stack S stores some integer values. Write a program that displays the integers from the stack S in sorted order with the help of a dequeue.*

   Use output-restricted deque D to accomplish this task as per following algorithm :

```
pop from stack, call it anum
if deque empty add to deque
if deque not empty
    if anum < front-most of deque
        add anum in beginning of deque
    otherwise
        add anum in the end of deque
repeat above steps for all elements of stack
finally display the deque
```

12

Simple Queries in SQL

rogress in SQL

12.1 Querying

First Things First

This practical session requires you to practise SQL statements. To practise these statements, you need help from your teacher as (s)he will create a blank database for you in the database software being used by your school, BEFORE you can start creating tables and running other queries.

12.1

Progress in SQL

Querying in SQL

Consider the tables :

(*a*) Customer

CUSTNUMB	CUSTNAME	ADDRESS	BALANCE	CREDLIM	SLSRNUMB
124	TINA ADAMS	481 Tilak Lane, CP, Delhi	41800.75	50000	3
256	R VENKAT	215 Mylapore, Chennai	100000.75	80000	6
311	PRAKASH DESHMUKH	48 Sea Link Raod, Mumbai	20000.10	30000	12
315	K R RAO	914 , K.R. Puram, Banglore	32000.75	30000	6
405	BOB MUKHERJEE	519 Salt Lake city, Kolkata	20100.75	80000	12
412	LUBNA ADAMS	16, Pedar Road, Mumbai	90800.75	100000	3
522	PRABHNOOR SINGH	108 CB, Vasant Kunj, Delhi	49000.50	80000	12
567	BHUVNA BALAJI	808, Bala Nagar, Hyderabad	20100.20	30000	6
587	JABBAR ALI	512 , Kokapet, Hyderabad	57000.75	50000	6
622	PRATHAM JAIN	419 , Plot 187, Sector-9, Dwarka, Delhi	57500.50	50000	3

(*b*) Order Details

ORDNUMB	PARTNUMB	NUMBORD	QUOTPRIC
12489	AX12	11	14.95
12491	BT04	1	402.99
12491	BZ66	1	311.95
12494	CB03	4	175.00
12495	CX11	2	57.95
12498	AZ52	2	22.95
12498	BA74	4	4.95
12500	BT04	1	402.99
12504	CZ81	2	108.99

(*c*) Orders

ORDNUMB	CUSTNO	ORDDTE
12489	124	01-MAR-18
12491	311	10-MAR-18
12494	315	31-MAR-18
12495	256	15-APR-18
12498	522	16-APR-18
12500	124	21-APR-18
12504	522	05-MAY-18

(*d*) Parts

PARTNUMB	PARTDESC	UNONHAND	ITEMCLSS	WREHSNM	UNITPRCE
AX12	IRON	104	HW	3	17.95
AZ52	SKATES	20	SG	2	24.95
BA74	BASEBALL	40	SG	1	4.95
BH22	TOASTER	95	HW	3	34.95
BT04	STOVE	11	AP	2	402.99
BZ66	WASHER	52	AP	3	311.95
CA14	SKILLET	2	HW	3	19.95
CB03	BIKE	44	SG	1	187.5
CX11	MIXER	112	HW	3	57.95
CZ81	WEIGHTS	208	SG	2	108.99

1. *Using the tables (Customer, OrderDetails, Parts, Orders) given above, type in the given commands, and in each case predict the output from each and compare the result with actual output.*

   For the given outputs, you have to write SQL statements to get that output.

   (*a*)
   ```
   Select custnumb, custname, address, balance from customer
   where credlim = 50000 ;
   ```

   Output

   (*b*) *What statement (SQL) would you write to get the following output from table Customer ?*

custnumb	custname	address	balance
622	PRATHAM JAIN	419 , Plot 187, Sector -9, Dwarka, Delhi	57500.50
587	JABBAR ALI	512 , Kokapet, Hyderabad	57000.75
124	TINA ADAMS	481 Tilak Lane, CP, Delhi	41800.75

(*c*) `Select custnumb, custname, address, balance from customer where credlim = 50000 order by custname ;`

Output

(*d*) `Select custnumb, custname, address, balance from customer where credlim = 50000 order by custnumb, custname;`

Output

(*e*) `Select custnumb, custname, address, balance from customer where credlim = 50000 order by custnumb desc, custname;`

Output

(*f*) What SQL statement would you write to get an output like the one shown below ?

custnumb	custname	address	balance
567	BHUVNA BALAJI	808, Bala Nagar, Hyderabad	20100.20
315	K R RAO	914 , K.R. Puram, Banglore	32000.75
311	PRAKASH DESHMUKH	48 Sea Link Raod, Mumbai	20000.10

(*g*) What SQL statement would you write to get an output like the one shown below ?

custnumb	*custname*	*address*	*balance*
124	TINA ADAMS	481 Tilak Lane, CP, Delhi	41800.75
315	K R RAO	914 , K.R. Puram, Banglore	32000.75
412	LUBNA ADAMS	16, Pedar Road, Mumbai	90800.75
622	PRATHAM JAIN	419 , Plot 187, Sector -9, Dwarka, Delhi	57500.50

(*h*)
```
Select custnumb, custname, address, balance from customer
where  custname > 'D';
```

Output

(*i*)
```
Select custnumb, custname, address, balance from customer
where  custname like 'ADAMS%' ;
```

(*j*) What SQL statement would you write to get an output like the one shown below ?

124	TINA ADAMS	481 Tilak Lane, CP, Delhi	41800.75
412	LUBNA ADAMS	16, Pedar Road, Mumbai	90800.75

Output

(*k*) `Select custnumb, custname, address, balance from customer where custname like '_ _N%';`

Output

(*l*) `Select custnumb, custname, address, balance from customer where custname like '%A%A%';`

Output

(*m*) `Select custnumb, custname, address, balance from customer where custname like '%A%I%';`

Output

2. *Given the following table called StuMarks having structure as :*

Stu_id	integer primary key
Student	varchar2(15) not null
Marks_1	Double
Marks_2	Double
Marks_3	Double

and following data :

Stu_id	StuName	Marks_1	Marks_2	Marks_3
11	Mikey Sharma	75.0	65.0	70.0
12	Josh Mark	80.0		60.0
13	Nigar Sultana	70.0	69.0	68.0
14	Ishpreet Kaur	69.0		70.0

(*a*) *Following SQL statement is not producing any output.*

Find out the problem, correct it and give the output produced.

```
select stu_id, marks_1, marks_2, marks_3 from stumarks
where marks_2 = null;
```

Problem

Corrected statement

Output

(*b*) *What will be the output produced by following statement ?*

```
select marks_2 from stumarks where marks_2 is not null;
```

(*c*) *Which SQL statement would give you following output ?*

12	Josh Mark	140
14	Ishpreet Kaur	139

(*d*) *What would be the output produced by following statement ?*

```
select stu_id, stuName, marks_1 + marks_2 + marks_3 from stumarks
where marks_2 is null;
```

(e) *Can you tell why there is difference between outputs of statements of parts (c) and (d) when basically both are doing the same thing ?*

3. *The following commands illustrate the logical test of values. Using the Part table (given above), do as directed.*

(a) *Which SQL command would give you following output ?*

(Hint : Make use of IN or NOT IN operators)

PARTNUMB	PARTDESC	UNONHAND	WREHSNM	UNITPRCE
BT04	STOVE	11	2	402.99
BZ66	WASHER	52	3	311.95

(b) *Which SQL command would give you following output ?*

(Hint : Make use of IN or NOT IN operators)

PARTNUMB	PARTDESC	UNONHAND	WREHSNM	UNITPRCE
AX12	IRON	104	3	17.95
AZ52	SKATES	20	2	24.95
BA74	BASEBALL	40	1	4.95
BH22	TOASTER	95	3	34.95
CA14	SKILLET	2	3	19.95
CB03	BIKE	44	1	187.5
CX11	MIXER	112	3	57.95
CZ81	WEIGHTS	208	2	108.99

(c) *Which SQL command would give you following output ?*

(Hint : Make use of BETWEEN operator)

PARTNUMB	PARTDESC	UNONHAND	WREHSNM	UNITPRCE
BZ66	WASHER	52	3	311.95
CB03	BIKE	44	1	187.5
CZ81	WEIGHTS	208	2	108.99

(d) *Which SQL command would give you following output ?*

(Hint : Make use of BETWEEN operator)

PARTNUMB	PARTDESC	UNONHAND	WREHSNM	UNITPRCE
CA14	SKILLET	2	3	19.95

4. *Carefully read following commands and predict the output and then run the statement and compare results.*

(a)
```
Select partnumb, partdesc, unonhand, wrehsnm, unitprce from parts
where itemclss = 'HW' and unitprce > 30.00;
```

(*b*) `Select partnumb, partdesc, unonhand, wrehsnm, unitprce from parts where itemclss = 'HW' or unitprce > 30.00;`

5. *The following commands illustrate the combining of commands using* ***and*** *or* ***or*** *operators. Using the PART table, do as directed :*

(*a*) *Which SQL command would give you following output ?*

(Hint : Make use of logical operators)

PARTNUMB	PARTDESC	UNONHAND	WREHSNM	UNITPRCE
AX12	IRON	104	3	17.95
BH22	TOASTER	95	3	34.95
CA14	SKILLET	2	3	19.95
CB03	BIKE	44	1	187.5
CX11	MIXER	112	3	57.95
CZ81	WEIGHTS	208	2	108.99

(*b*) *Which SQL command would give you following output ?*

(Hint : Make use of logical operators)

PARTNUMB	PARTDESC	UNONHAND	WREHSNM	UNITPRCE
AZ52	SKATES	20	2	24.95
BA74	BASEBALL	40	1	4.95

PARTNUMB	PARTDESC	UNONHAND	WREHSNM	UNITPRCE
CB03	BIKE	44	1	187.5
CX11	MIXER	112	3	57.95
CZ81	WEIGHTS	208	2	108.99

(c) *Which SQL command would give you following output ?*

(**Hint** : Make use of logical operators)

PARTNUMB	PARTDESC	UNONHAND	WREHSNM	UNITPRCE
AZ52	SKATES	20	2	24.95
BA74	BASEBALL	40	1	4.95
CB03	BIKE	44	1	187.5
CX11	MIXER	112	3	57.95
CZ81	WEIGHTS	208	2	108.99

(d) *Which SQL command would give you following output ?*

(**Hint** : Make use of logical operators)

PARTNUMB	PARTDESC	UNONHAND	WREHSNM	UNITPRCE
CB03	BIKE	44	1	187.5
CX11	MIXER	112	3	57.95
CZ81	WEIGHTS	208	2	108.99

13

Table Creation and Data Manipulation Commands

Progress in SQL

First Things First

This practical session requires you to practise SQL statements. To practise these statements, you need help from your teacher as (s)he will create a blank database for you in the database software being used by your school, BEFORE you can start creating tables and running other queries.

13.1

Progress in SQL

Creating Tables

1. *Following table represents information on sales representatives for Premiere Products, and contains the following data fields for sales representatives :*

   sales representative's number as SLSRNUMB,
   sales representative's name as SLSRNAME,
   sales representative's address as SLSADDR,
   total commission paid to the sales representative as TOTCOMM, and
   sales representative's commission rate as COMMRATE.

Sales Representative

SLSRNUMB	SLSRNAME	SLSRADDR	TOTCOMM	COMMRATE
3	ANUSHA JAIN	BH 130, Shalimar Bagh, N. Delhi	2150.00	0.05
6	KULWANT SINGH	K 2130, Vaishali, N. Delhi	4912.00	0.07
12	SAM BROWN	310, Behtar Appts, Sec-13, Rohini, Delhi	2150.00	0.05

(*a*) *Write command to create the above table by typing following statement :*

```
create table SALESREP (
slsrnumb integer,
slsrname varchar(25) not null,
slsraddr varchar(100) not null,
totcomm double,
commrate double
);
```

Write in box given below, the response returned by the database software after running above statement :

(*b*) *To see what you have created, i.e., the structure of the table, type the following command :*

```
describe salesrep;
```

Write in below box the response returned by the database software after running above statement :

- You must have observed that **Describe** command lists the structure of a table that has been created earlier.

2. *Give following* ***Create Table*** *commands as given below and list their table-structures.*

(*a*)
```
CREATE TABLE LOCATION
(
 id            NUMERIC(5)    PRIMARY KEY,
 bldg_code     VARCHAR(10)   NOT NULL,
 room          VARCHAR(6)    NOT NULL,
 capacity      NUMERIC(5)    NOT NULL
);
```

Table Structure :

(*b*)
```
CREATE TABLE faculty
(
id             NUMERIC(5)    PRIMARY KEY,
lastName       VARCHAR(30)   NOT NULL,
firstName      VARCHAR(30)   NOT NULL,
locationid     NUMERIC(5)    REFERENCES location(id),
phone          VARCHAR(10),
rank           VARCHAR(8)
CHECK ((rank = 'ASSO') OR (rank = 'FULL') OR (rank = 'ASST')
OR (rank = 'INST')),
startdate DATE
);
```

Table Structure :

(*c*)
```
CREATE TABLE student
(
   id          NUMERIC(5)    PRIMARY KEY,
   lastName    VARCHAR(30)   NOT NULL,
   firstName   VARCHAR(30)   NOT NULL,
   street      VARCHAR(25),
   city        VARCHAR(20),
   state       CHAR(5)       DEFAULT 'Del',
   zip         VARCHAR(9),
   phone       VARCHAR(10),
   class       CHAR(2)       DEFAULT 'FR',
   dob         DATE,
   facultyid     NUMERIC(4) REFERENCES faculty(id)
);
```

Table Structure :

(*d*)
```
CREATE TABLE TERM
(
description    VARCHAR(20)    PRIMARY KEY,
status         VARCHAR(20)    NOT NULL
CHECK ((status = 'OPEN') OR (status = 'CLOSED'))
);
```

Table Structure :

(*e*)
```
CREATE TABLE COURSE
(
id      NUMERIC(6)    PRIMARY KEY,
code    VARCHAR(10)   NOT NULL,
name    VARCHAR(25),
credit NUMERIC(2)     DEFAULT 3
);
```

Table Structure :

3. *List the constraints in each of the tables created in previous question and give their type, i.e., whether a column constraint or a table constraint.*

(*a*) **Location** Table's constraints

(i)

(*b*) **Faculty** Table's constraints

(i)

(ii)

(iii)

(*c*) **Student** Table's constraints

(i)

(ii)

(iii)

(iv)

(*d*) **Term** Table's constraints

(i)

(ii)

(*e*) **Course** Table's constraints

(i)

(ii)

13.2

Progress in SQL

Inserting Data in Tables

1. *The Insert Into < Table> command lets you insert records in existing tables. For the tables that you created in* ***PriS18.1****, it's time you added records to them. So enter following Insert Into commands :*

(*a*) Inserting in Salesrep table

```
insert into salesrep values (3,'ANUSHA JAIN','BH 130, Shalimar Bagh,
N. Delhi',2150.00,0.05);
insert into salesrep values (6,'Kulwant Singh','K 2130,  Vaishali ,
N. Delhi', 4912.00, 0.07);
insert into salesrep values (12,'Sam Brown','310, Behtar Appts, Sec-13,
Rohini, Delhi', 2150.00, 0.05);
```

(*b*) Inserting in Location table

```
Insert into Location Values (53, 'BUS', '424', 1 );
Insert into Location Values ( 54, 'BUS', '402', 1);
Insert into Location Values ( 45, 'CR', '101', 150);
Insert into Location Values ( 46, 'CR', '202', 40);
Insert into Location Values ( 47, 'CR', '103', 35);
Insert into Location Values ( 48, 'CR', '103', 35);
Insert into Location Values ( 49, 'BUS', '105', 42);
Insert into Location Values ( 50, 'BUS', '404', 35);
Insert into Location Values ( 51, 'BUS', '421', 35);
Insert into Location Values ( 52, 'BUS', '211', 55);
Insert into Location Values ( 55, 'BUS', '433', 1);
Insert into Location Values ( 56, 'LIB', '217', 1);
Insert into Location Values ( 57, 'LIB', '222', 1);
```

(*c*) Inserting in Faculty table

(Notice that Date( ) has been used to convert a string date in form 'yyyy-mm-dd' to ***date*** *type)*

```
INSERT INTO faculty VALUES
(1, 'Sharma', 'Kiran', 53, '5551234', 'ASSO', date('1995-09-15') );
INSERT INTO faculty VALUES
(2, 'Bhattcharya','Jiten', 54, '5559087', 'FULL', date('1972-01-12'));
INSERT INTO faculty VALUES
(3, 'Williams', 'Jerry', 56, '5555412', 'ASST', date('1992-08-26'));
INSERT INTO faculty VALUES
(4, 'Narayan','Lalit', 55, '5556409', 'INST',  date('1995-01-22'));
INSERT INTO faculty VALUES
(5, 'Khan', 'Abdul', 57, '5556082', 'ASSO',  date('1985-08-15'));
```

(*d*) Inserting records into **Student** table

```
INSERT INTO student VALUES
(100, 'Mittal', 'Suhani', '144 AB Street', 'Mumbai', 'Mah', '400001',
'5559876','SR', Date('1999-07-14'), 1);
INSERT INTO student VALUES
(101, 'Robinson', 'Brian', '454 K Nagar' , 'Delhi', 'Del', '110001',
'5552345', 'SR', Date('1999-08-19/'), 1);
INSERT INTO student VALUES
(102, 'Merchant', 'Danish', '8921 Salt Lake City', 'Kolkata', 'WB', '547105',
'5553907','JR', Date('1997-10-10'), 1);
INSERT INTO student VALUES
(103, 'Mojumdar', 'Anusha', '1716 Kalighat.', 'Kolkata', 'WB', '547003',
'5556902', 'SO', Date('1998-09-24'), 2);
INSERT INTO student VALUES
(104, 'Swami', 'Rajendran', '1780 T. Nagar', 'Chennai', 'Tam', '647001',
'5558899', 'SO', Date('1997-11-20'), 4);
INSERT INTO student VALUES
(105, 'Kaur', 'Simran', '1818 Vasant Kunj', 'Delhi', 'Del', '110071',
'5554944', 'FR', Date('1997-12-4'), 3);
```

(*e*) Inserting records into **Term** table

```
INSERT INTO term VALUES  ('Term1 2013', 'CLOSED');
INSERT INTO term VALUES  ('Term1 2014', 'CLOSED');
INSERT INTO term VALUES  ('Term2 2014', 'OPEN');
INSERT INTO term VALUES  ('Term3, 2014', 'OPEN');
INSERT INTO term VALUES  ('Term1 2014', 'OPEN');
INSERT INTO term VALUES  ('Term2 2015', 'OPEN');
INSERT INTO term VALUES  ('Term3 2015', 'OPEN');
```

(*f*) Inserting records into **Course** table

```
INSERT INTO course VALUES
(1, 'MIS 101', 'Intro. to Info. Systems', 3);
INSERT INTO course VALUES
(2, 'MIS 301', 'Systems Analysis', 3);
INSERT INTO course VALUES
(3, 'MIS 441', 'Database Management', 3);
INSERT INTO course VALUES
(4, 'CS 083', 'Programming in C++', 3);
INSERT INTO course VALUES
(5, 'CS 083N', 'Programming in Python', 3);
```

2. *To see what all records are there in a table, you can write following statement* :

```
SELECT * FROM <table name> ;
```

That is to view records of Salesrep table, you will write :

```
SELECT * FROM Salesrep;
```

Based on this, issue commands to view records of Faculty, Student, Term and Course Tables.

(*a*) Command for Listing records of Location table

(*b*) Command for Listing records of Faculty table

(*c*) Command for Listing records of Student table

(*d*) Command for Listing records of Term table

(*e*) Command for Listing records of Course table

3. *Create tables ((Customer, OrderDetails, Parts, Orders) given in PriS 17.1. Use create table and INSERT INTO <table> commands.*

(*a*) *Create Table Customers*

Insert Commands

(*b*)

Create Table *OrderDetails*

Insert Commands

(*c*)

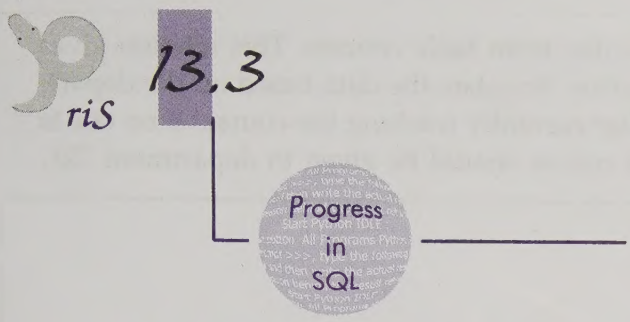
Create Table *Parts*

Insert Commands

(*d*)

Create Table **Orders**

Insert Commands



DDL Commands

1. *Given a database with following tables :*

   Table : **Dept**

   `deptId (integer), deptName (varchar(40)), and deptLocation (varchar(50)).`

   Table : **courseDept**

   `deptId (integer) and cname (varchar(40)).`

   Table : **faculty**

   `fid (Varchar (10)), fname (Varchar (40)), basicPay (double), deptId (integer)`

   Perform given tasks on these tables.

   (*a*) Delete the student with number = '318548912'.

   (*b*) Change the name of student 'Anusha John' to 'Anushka John'. Update her age to be 20 as well.

   (*c*) Change the location of department with 33 to 'D Block, 23FF'.

   (*d*) Increase basicPay of all faculty members by 25%.

   (*e*) Change the name of the course 'Database Systems' to 'Introduction to Database Systems'.

(*f*) Add tuples to the courseDept relation from table courses. This relation gives what departments teach what courses. Populate the data based on the department number of the faculty member currently teaching the course. If no one is currently teaching the course, the course should be given to department '20'.

2. *Consider the faculty table given in* ***previous Question 1.***

(*a*) Modify the faculty table so that the width of **fname** is increased to 50 and it ensures that no NULL or empty value is entered to it.

(*b*) Modify the faculty table so that the **deptId** is a foreign key to the **dept** table.

3. *Consider database given in* ***previous Question 1****. Write SQL statements to remove everything (tables etc) from the database.*

Progress in
COMPUTER SCIENCE
with python

Computer Science with Python (083) is a major subject at Senior Secondary level of CBSE. This practical book is the component book for the same. This book contains practical exercises to help the learner with concepts of Computer Fundamentals and Python.

At various points in the textbook chapters, *Progress in Python (PriP)* sessions have been suggested. If the students do it in the same way, it will help them learn and understand concepts thoroughly. Therefore, it is highly recommended to do these inclusive practical sessions along with chapters.

In our pilot-run of the book, it was assessed that inclusive practice of these practical sessions along with textbook ensured better learning and understanding of concepts than just the textbook.

We are sure that this book will meet your expectations and requirements fully.

PUBLISHER

Published by
DHANPAT RAI & CO. (P) LTD.
Educational & Technical Publishers
4576/15, Agarwal Road, Darya Ganj, New Delhi-110002
Ph.: 2324 7736, 37, 38 • E-mail: dhanpatrai@gmail.com

ISBN 978-81-7700-236-2