EX.NO : 01

PYTHON PROGRAM FOR INPUT AND OUTPUT FUNCTIONS

a) PROBLEM STATEMENT :

Rahul gets multiple inputs from user as list 1 and list 2. His task is to perform concatenate operation between two lists using slice operator displaying the index values from 1 to 3. Display the values of both the list in append method using for loop.

AIM:

To perform concatenation slicing operations and Duppo append method using for loop in the input lists.

ALGORITHM :

Step 1 : Start the program.

Step 2 : Get the two list as inputs.

Step 3 : Concatenate the two list and print the concatenated list.

Step 4 : Slice the list.

Step 5 : Use append method using for loop to print the list.

Step 6 : Stop the program.

2. PROBLEM STATEMENT:

Write a python program for 'iterative approach' using input and output function.

AIM:

To perform 'iterative approach' using input and output function.

ALGORITHM:

Step 1: Start the program.

Step 2: Get the input values for list.

Step 3: Use for loop to print each values in the list.

Step 4: Stop the program.

C) PROBLEM STATEMENT:

Write a python program to find the given integer in the given list of integers and print the number of occurances of that integer in the list.

AIM:

To find the number of occurances of that integer in the list.

ALGORITHM:

Step 1: Start the program.

Step 2: Assign some values as input to the list.

Step 3: Get the integer which is to be founded

Step 4: Increase the count which is to be ~~four~~ initialised as zero, if integer is found in the given list.

Step 5: If not found, print that element is not found.

Step 6: Stop the program.

Ex. NO : 02

PYTHON PROGRAM USING CONTROL FLOW STATEMENT AND FUNCTIONS

a) PROBLEM STATEMENT:

Write a python program to check leap year using if else.

AIM:

To find whether the given year is leap year or not using 'if else' condition.

ALGORITHM:

Step 1 : Start the program.

Step 2 : Take the input as year format.

Step 3 : print 'leap year' if the given year is divided by 4, 400 and 100 using modulo operator ie, (year % 4 == 0).

Step 4 : print 'not a leap year' if the condition is false.

Step 5 : Stop the program.

b) PROBLEM STATEMENT :

Write a python program to find the fibonacci series using while loop.

AIM:

TO find the fibonacci series using while loop.

ALGORITHM :

Step 1: Start the program.

Step 2: Take the input.

Step 3: Initialize i=0, first = 0 and second = 1.

Step 4: Use 'while' loop to check the conditions if i<a, it enter the 'while' loop else, it comes out of the loop.

Step 5: If i≤i, the output is 'i' itself, else it enters into the next loop 'else'.

Step 6: In 'else' loop, the first two succeeding numbers are added and stored in 'next' variable. Then assigning 'first = second' and 'second = next'. it continues until satisfying the while condition, atlast, the fibonacci series is formed.

Step 7 : Stop the program.

c) PROBLEM STATEMENT :

Write a python program to display the following pattern using nested - for loop.

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

AIM :

To display the pattern using nested for loop.

ALGORITHM :

Step 1 : Start the program.

Step 2 : Get the rows value as 5 at the run time.

Step 3 : Use 'for' loop for iterating through the rows until it reaching 5.

Step 4 : Inside the 'for' loop use another 'for' loop as nested loop for iterating through columns until it reaches 5. use 'end' at print line.

Step 5 : Stop the program.

d) PROBLEM STATEMENT :

Write a python program to find factorial of a number using recursive function in python.

AIM:

To find the factorial of a number using recursive function.

ALGORITHM :

Step 1 : Start the program.

Step 2 : Use define function. If 'n==1' and 'n==0' return n as output else return n* fact (n-1) as output.

Step 3 : Get the input.

Step 4 : Print error message if input is not valid ie less than 0.

Step 5 : print the output for the valid input using function calling.

Step 6 : Stop the program.

e) PROBLEM STAMENT :

Create a function show Employee () in such a way that it should accept employee id, employee name, destination and salary and display all. If the salary is missing in the function call. assign default value 9000 to salary.

AIM :

To display the employee details and employee salary as default if the input is missing.

ALGORITHMS :

Step 1 : Start the program.

Step 2 : Use 'define' function inside that functions assign the arguments.

Step 3 : Within the 'def' function, print the employee details. if the function is called.

Step 4 : Get the employee details as input.

Step 5 : while getting employee salary, assign default value as '9000' using 'or' operator along with input function.

Step 6 : Stop the program.

Ex. No : 03

PYTHON PROGRAM TO IMPLEMENT VARIOUS OPERATIONS ON STRING

a) PROBLEM STATEMENT :

Write a python program which contains minimum 25 string function in it.

AIM :

To perform minimum 25 string functions.

ALGORITHMS :

Step 1 : Start the program.

Step 2 : Get the input string.

Step 3 : Use 'capitalize()' to capitalize the first letter of the sentence.

Step 4 : Use 'center()', to print the given string at the center.

Step 5 : Use 'count()', to count the occurances of the particular character at the given string.

Step 6 : Use 'endswith()', to check the end of the string.

Step 7 : Use 'find()', to find the particular character in the given string.

Step 8 : Use 'index()', to find the index of the particular character in the given string.

Step 9 : Use 'isalnum()', to check whether alphanumeric or not.

Step 10 : Use 'isascii()', to check whether ascii or not.

Step 11 : Use 'isdecimal()', to check whether decimal or not.

Step 12 : Use 'isdigit()', to check whether digit or not.

Step 13 : Use 'isidentifier()', to check whether identifier or not.

Step 14 : Use 'islower()', to check lowercase or not.

Step 15 : Use 'isnumeric(), to check numeric number or not.

Step 16 : Use 'isprintable()', to check printable or not.

Step 17 : Use 'istitle()', to check title or not.

Step 18 : Use 'isupper()', to check uppercase or not.

Step 19 : Use 'join()'. To join the individual character in the given string.

Step 20 : Use 'ljust()' and 'rjust()', to print special characters ~~in~~ at the ~~given~~ left and right side.

Step 21 : Use 'replace()', to replace old characters with the new character in given string.

Step 22 : Use 'strip()', to remove unwanted spaces.

Step 23 : Use 'partition()', to separate the string with mentioned character at centre in tuple format.

Step 24 : Use 'split()', to separate the string as partition function, but in form of list.

Step 25 : Use 'swapcase()', to convert lowercase into uppercase and vice-versa.

Step 26 : Use 'upper()', to capitalize all the characters in the string.

Step 27 : Use 'casefold()', to lowercase the uppercased letters.

Step 28 : Stop the program.

b) PROBLEM STATEMENT:

User ID Generation.

Joseph's team has been assigned the task of Creating users-lists for all participants of an online gaming competition Joseph has designed a process for generating the user-id using the participant First_name, last_name, PIN_Code and number N. The process defined by Joseph is as below:

Step 1: Compare the lengths of First name and Last_name of the participants. The one that is shorter will be called 'smaller name' and longer called 'longer name'. If both are in equal length, then the name that appears earlier in alphabetical order will be 'smaller name' and name appears later called 'Longer name'.

Step 2: the user-id should be generated as below First letter of longer name + entire word of second name + digit at position N in PIN when the transversing the PIN from left to right + digit at position N in PIN when transversing the PIN from right to left.

Step 3: Toggle the alphabets of user-id generated in Step-2 is uppercase alphabets should become lowercase and lowercase should become uppercase.

AIM:

To write a program for user id generation.

ALGORITHM:

Step 1: Start the program.

Step 2: Get the first name, last name, pin no and N values as input.

Step 3: Convert the pin, from integer to string datatype.

Step 4: Reverse the pin.

Step 5: Take longer name as first name, smaller name as last name. If length of ~~long~~ first name is greater than last name and vice-versa.

Step 6: Use min() and max(), if both the firstname and lastname have same length.

Step 7: Use indexing and concatenation operations inorder to the user id generation.

Step 8: Stop the program.

Ex. NO : 04

PYTHON PROGRAM TO IMPLEMENT VARIOUS OPERATIONS ON LIST

a) PROBLEM STATEMENT :

Write a python program to solve all list methods in python.

AIM :

To perform all the list operations.

ALGORITHM :

Step 1 : Start the program.

Step 2 : Get the list as input along with that assign the size.

Step 3 : Use 'append()' method to get the list elements.

Step 4 : Use 'copy()', to copy the list elements.

Step 5 : Use 'count()', to count the occurance of the particular elements.

Step 6 : Use 'extend()', to extend the size of the list

Step 7 : Use 'pop()', to remove element based on index.

Step 8 : Use 'remove()', to remove the particular element.

Step 9 : Use 'insert()', to insert the particular element at the particular index.

Step 10 : Use 'sort()', to sort the elements in the list at a particular order.

Step 11 : Use 'clear()', to clear all elements in the list.

Step 12 : Use 'reverse()', to reverse the element in the list.

Step 13 : Stop the program.

b) PROBLEM STATEMENT :

A Certain business maintains a list of its customer's names. The list is arranged in order of importance with last customer in list being most import and Now, he want to create a new list sorted alphabetically according to customers last names. but among customers with same last name he want the most important ones to apper earlier in the new list alphabetically order (equality of last name) should not be case sensitive.

Input :

First line contains no. of test cases and first line of each test case contains n i.e. no. of element and next n lines contains a name.

OUTPUT :

print the new list with each element in a new line.

AIM:

To perform sorting operation based on customers last name.

ALGORITHM:

Step 1 : Start the program.

Step 2 : Get the list count as input.

Step 3 : Get the size of the list to be created.

Step 4 : Use 'for loop' to get the list elements.

Step 5 : Use 'sort()', to perform the sorting operation.

Step 6 : use 'lambda()', the last elements based on the last names.

Step 7 : Split the names, arrange the names using d[0],d[1] = d[1], d[0].

Step 8 : Use 'join()', to join the divided names.

Step 9 : Stop the program.

Ex.NO: 05	PYTHON PROGRAM TO IMPLEMENT VARIOUS OPERATIONS ON SETS

PROBLEM STATEMENT:

Write a python program to implement various operations on set.

AIM:

To implement various operations on set.

ALGORITHM:

Step 1 : Start the program.

Step 2 : Get the input as set 1 and set 2 at the run time.

Step 3 : Use len(). to find out the length of input.

Step 4 : Use type(), to find out type of input.

Step 5 : Use membership operator inorder to check the occurance of particular character.

Step 6 : Use add() to add particular value. remove() to remove particular value.

Step 7 : Use update() to update one set into another set.

Step 8 : Use intersecton_update() to insert the common elements into another set.

Step 9 : Use union(), to unite the two sets.

Step 10 : Use symmetric_difference_update(), the elements from the set 1 will be printed eliminating the elements from the set 2.

Step 11 : Stop the program.

EX.NO:06	PYTHON PROGRAM TO IMPLEMENT VARIOUS OPERATIONS ON DICTIONARY

PROBLEM STATEMENT :

Write a python program to implement Various operations on dictionary.

AIM :

To implement various operations on dictionary.

ALGORITHM :

Step 1 : Start the program.

Step 2 : Get the two dictionaries as input.

Step 3 : Use 'len()', to print length of given dictionary.

Step 4 : Use items(), to print keys of the dictionary.

Step 5 : Use 'keys()', to print keys of the dictionary.

Step 6 : Use 'values()', to print values of the dictionary.

Step 7 : Use 'copy()', to copy the dictionary items.

Step 8 : Use 'clear()', to clear dictionary items.

Step 9 : Use 'fromkeys()', to assign particular value to all key in dictionary.

Step 10 : Use 'setdefault()', to assign default value.

Step 11 : Stop the program.

EX.NO:07	PYTHON PROGRAM TO IMPLEMENT VARIOUS OPERATIONS ON TUPLES

a) PROBLEM STATEMENT:

Write a python program to implement various operations on tuples.

AIM:

To implement various operations on tuple.

ALGORITHM:

Step 1 : Start the program.

Step 2 : Use index(), to find out index of particular value.

Step 3 : Use count(), to find no. of occurance of the particular value.

Step 4 : Use concatination operation to add two tuples.

Step 5 : Use len() to find out the length of tuple.

Step 6 : Use '*' operator inorder to create the copies of the elements in the tuple.

Step 7 : Stop the program.

b) PROBLEM STATEMENT:

Write a program to perform addition operation of elements in tuple with and without 'for loop'.

AIM:

To perform addition operation of elements in tuple with and without 'for loop'.

ALGORITHM:

Step 1 : Start the program.

Step 2 : Get the tuple as input at run time.

Step 3 : Use sum() to sum the addition of elements in the given tuple without using 'for loop'.

Step 4 : Assign temp = 0, then use 'for loop' for iterating the individual elements in the tuple.

Step 5 : Through iterating, each element will be added to the temp.

Step 6 : Store the value in temp i.e, sum of elements.

Step 7 : Stop the program.