Experiment No:1 Lab Cycle : 01 Date : 09-09-2024

LEAP YEAR

AIM:

Write a python program to display future leap years from current year to a final entered by user.

ALGORITHM:

```
1. Start.
2. Input: Enter the final year (year).
3. Initialize: Create an empty list leap.
4. Iterate: Start a loop from 2024 to year (inclusive).
        a.  If the year is divisible by 400, it is a leap year.
        b. Else if the year is divisible by 4 and not divisible by
        100, it is a leap year.
        c. If any of the above conditions are true, add the year to the leap list.
5. Display result
6. Stop.
```

SOURCECODE:

```
year=int(input("Enter the final year"))
leap=[]
for i in range(2024,year+1):
        if i%400==0 or i%4==0 and i%100!=0:
                leap.append(i)
print("LEAP YEARS :",leap)
```

OUTPUT:

```
Enter the final year: 2032
LEAP YEARS: [2024, 2028,2032]
```

RESULT:

The program ran successfully and output is verified.

Experiment No:2.1 Lab Cycle:01 Date : 09-09-2024

POSITIVE LITS OF NUMBERS

AIM:

Write a python program to generate positive list of numbers from a given list of numbers

ALGORITHM:

```
1. Start
2. Input: Provide a list of numbers (numbers).
3. Initialize: Create an empty list pos to store positive numbers.
4. Iterate: For each number i in the list numbers:
    a. If i is greater than 0, it is a positive number.
    b. Append i to the list pos.
5. Output: Display the list pos containing all positive numbers.
6. Stop..
```

SOURCE CODE:

```
list=[1,-2,3,-1,4,-6,9,-7,5]
pos=[]
for i in list:
    if i >0:
        pos.append(i)
print("Positive numbers : ",pos)
```

OUTPUT:

```
list = [1, -2, 3, -1, 4, -6, 9, -7, 5]

Positive numbers: [1, 3, 4, 9, 5]
```

RESULT:

The program ran successfully and output is verified.

Experiment No:2.2 Lab Cycle:01 Date:10-10-2024

SQUARE OF N NUMBERS

AIM:

Write a Python program to generate Square of N numbers.

ALGORITHM:

```
1. Input the number of elements n
2. Initialize an empty list to store the numbers.
3. Loop from 1 to n. For each iteration: Prompt the user to enter a number, Append the entered number to the list.
4. Compute the squares by using a loop to calculate the square of each number in the list and store the squared values in a new list.
5. Print the result.
```

SOURCE CODE:

```
n = int(input("Enter the number of elements to square: "))
numbers = []
for i in range(n):
num = int(input("Enter number " + str(i + 1) + ": "))
numbers.append(num)
squared_numbers = []
for x in numbers:
squared_numbers.append(x * x)
print("Original numbers:", numbers)
print("Squares of the numbers:", squared_numbers)
```

OUTPUT:

```
Enter the number of elements to square: 3
Enter number 1: 2
Enter number 2: 5
Enter number 3: 3
Original numbers: [2, 5, 3]
Squares of the numbers: [4, 25,
```

RESULT:

The program ran successfully and output is verified.

Experiment No:2.3 Lab Cycle:01 Date:10-10-2024

LIST OF VOWELS IN A WORD

AIM:

Write a Python program to find the list of vowels in a word

ALGORITHM:

```
1. Start.
2. Input Word: Prompt the user to enter a word.
3. Find Vowels:
   Create an empty list list1.
   For each character in the word, if it is a vowel and not already in list1, add it to list1.
6. Output: Print the vowels stored in list1.
5. Stop.
```

SOURCE CODE:

```
element=input("Enter a word:")
vowels=['a','e','i','o','u']
list1=[]
for x in element:
 if(x in vowels and x not in list1):
     list1.append(x)
print("vowels present in the given word are: ",list1)
```

OUTPUT:

```
Enter a word: english
vowels present in the given word are: ['e', 'i']
```

RESULT:

The program ran successfully and output is verified.

Experiment No:2.4 Lab Cycle:01 Date:10-10-2024

ORDINAL VALUE

AIM:

Write a Python program to find the list of ordinal valuein a word

ALGORITHM:

```
1. Start.
2. Input Word: Prompt the user to enter a word and store it in word.
3.Iterate through the Word:
    For each character i in the word,
    Find the ordinal value of the character using ord(i).
    Print the character and its ordinal value.
4. Stop.
```

SOURCE CODE:

```
word = input("Enter word : ")
for i in word:
    print("Ordinal value of ",i," is :",ord(i))
```

OUTPUT:

```
Enter word : hello
Ordinal value of h is : 104
Ordinal value of e is : 101
Ordinal value of l is : 108
Ordinal value of l is : 108
Ordinal value of o is : 111
```

RESULT:

The program ran successfully and output is verified.

Experiment No: 3 Lab Cycle:01 Date:10-10-2024

OCCURRENCE OF THE WORD

AIM:

Count the occurence of each word in a line of text.

ALGORITHM:

```
1. Start.
2. Input: Take a line of text from the user (str).
3. Initialize:
      An empty dictionary freq to store word occurrences.
4. Split the string: Divide the string into words using the split() function.
5. Iterate: For each word in the list of words:
      a. If the word is already in the dictionary freq, increment its value by 1.
      b. If the word is not in the dictionary, add it with a value of 1.
6. Output: Display each word and its count from the dictionary freq.
7. Stop.
```

SOURCE CODE:

```
str = input("Enter a string")
freq = {}

for char in str:
    if char in freq:
        freq[char] += 1
    else:
        freq[char] = 1

for char, count in freq.items():
    print(f"{char}: {count}")
```

OUTPUT:

```
Enter the string: hello hai how are you hello hai

Word occurrences:
hello: 2
hai: 2
how: 1
are: 1
you: 1
```

RESULT:

The program ran successfully and output is verified.

Experiment No:4 Lab Cycle:01 Date:10-10-2024

FILTERING INTEGERS

AIM:

Prompt the user for a list of integers. For all values greater than 100, store ‘over’ instead.

ALGORITHM:

```
1. Start Start.
2. Input: Prompt the user to enter a list of integers separated by spaces.
3. Convert Input:
   Split the input string into a list of substrings using split().
   Convert each substring into an integer and store the result in a list number.
4. Process the List:
   Use a list comprehension to iterate through the list number.
   For each integer num in number:
   If num is greater than 100, replace it with the string 'over'.
   Otherwise, keep the integer as it is.
   Store the processed results in a new list result.
5. Output: Print the processed list result.
6. Stop.
```

SOURCE CODE:

```
user_input = input(“Enter a list of integers separated by spaces: ”)

number =[int(num) for num in user_input.split()]

result = [‘over’ if num>100 else num for num in number]

print(“Processed list : ”,result)
```

OUTPUT:

```
Enter a list of integers separated by spaces: 34 56 102 78 123
Processed list : [34, 56, 'over', 78, 'over']
```

RESULT:

The program ran successfully and output is verified.

Experiment No: 5 Lab Cycle:01 Date:10-10-2024

COUNT THE OCCURRANCE OF 'A'

AIM:

Store a list of first names. Count the occurences of 'a' within the list.

ALGORITHM:

```
1. Start.
2. Input: Ask the user to input words for the list, separated by spaces.
3. Initialize:
      Convert the user input into a list using the split() function.
      Set a variable out to 0 to store the total count of the letter 'a'.
4. Iterate: For each word i in the list:
     Convert the word to lowercase using the lower() method.
     Count the occurrences of the letter 'a' in the word using the count("a") method.
     Add the count to out.
5. Output: Display the total number of occurrences of 'a' in all the words.
6. Stop.
```

SOURCE CODE:

```
words = input("Enter the words separated by spaces: ").split()

out = 0

for word in words:
    out += word.lower().count("a")

print("The total number of 'a' in the words is:", out)
```

OUTPUT:

```
Enter the words separated by spaces: amal Arun aman arjun Anwar

The total number of 'a' in the words is: 8
```

RESULT:

The program ran successfully and output is verified.

Experiment No: 6 Lab Cycle:01 Date:10-10-2024

LIST OF INTEGERS

AIM:

Enter 2 lists of integers , Check:
(a) whether lists are of same length
(b) whether both lists sums to same value
(c) whether any value occur i both

ALGORITHM:

1. Input two list of integers
2. Check whether lists are of same length
3. Check whether both the list sums to same value
4. Check whether any value occur in both

SOURCE CODE:

```
l1 = list(map(int, input("Enter the elements of the first list separated by spaces: ").split()))
l2 = list(map(int, input("Enter the elements of the second list separated by spaces: ").split()))

if len(l1) == len(l2):
    print("Both lists have the same length.")
else:
    print("Different lengths.")

if sum(l1) == sum(l2):
    print("Both lists have the same sum.")
else:
    print("Different sums.")

common_elements = []
for i in l1:
    if i not in common_elements and i in l2:
        common_elements.append(i)

print("The common elements are:", common_elements)
```

OUTPUT:

```
Enter the elements of the first list separated by spaces: 1 2 3 4 5
Enter the elements of the second list separated by spaces: 7 8 9 10 4

Different lengths.
Different sums.
The common elements are: [4]
```

RESULT:

The program ran successfully and output is verified.

Experiment No:07 Lab Cycle:01 Date:10-10-2024

REPLACING FIRST CHARACTER WITH $

AIM:

Write a Python program to get a string from an input string where all occurrence of first character replaced with '$', except first character.

ALGORITHM:

```
1. Input a string, input_string
2. Extract the first character of input_string as first_char.
3. Initialize modified_string with first_char.
4. Iterate through the remaining characters of input_string:
    a)If the current character matches first_char (case-insensitive), append '$' to
      modified_string.
    b)Otherwise, append the current character as is.
5. Print modified_string.
```

SOURCE CODE:

```
input_string = input("Enter a string: ")
first_char = input_string[0]
modified_string = first_char
for char in input_string[1:]:
    if char.lower() == first_char.lower():
        modified_string += '$'
    else:
        modified_string += char
print("Modified string:", modified_string)
```

OUTPUT:

```
Enter a string: Onion
Modified string: Oni$n
```

RESULT:

The program ran successfully and output is verified.

Experiment No:8 Lab Cycle:01 Date:30-10-2024

EXCHANGING FIRST AND LAST

AIM:

Write a Python program to interchange the first and last character of a string.

ALGORITHM:

```
1.Start
2.Input string from the user.
3.Interchange first and last characters.
4.Print original and modified strings.
5.Stop
```

SOURCE CODE:

```python
string = input("Enter the string: ")
print("Original String:",string)
string = string[-1]+string[1:-1]+string[0]
print("String after interchange:",string)
```

OUTPUT:

```
Enter the string: python
Original String: python
String after interchange: nythop
```

RESULT:

The program ran successfully and output is verified.

Experiment No:9 Lab Cycle:01 Date:30-10-2024

AREA OF CIRCLE

AIM:

Write a Python program to accept the radius from user and find area of circle.

ALGORITHM:

```
1. Start.
2. Input the radius of the circle (radius) from the user.
3. Calculate the area and store it in a variable (area).
       area=3.14*radius*radius
4. Display the area of the circle.
5. Stop.
```

SOURCE CODE:

```
radius=float(input("Enter the radius of a circle:"))
area=3.14*radius*radius
print("The area of circle is :",area)
```

OUTPUT:

```
Enter the radius of a circle:5
The area of circle is : 78.5
```

RESULT:

The program ran successfully and output is verified.

BIGGEST OF 3 NUMBERS

AIM:

Write a Python program to find biggest of 3 numbers entered.

ALGORITHM:

```
1. Start.
2. Input three numbers (x, y, and z).
3. Compare the numbers using the following conditions:
   a. If x >= y and x >= z, assign biggest = x.
   b. Else if y >= x and y >= z, assign biggest = y.
   c. Else, assign biggest = z.
4. Display the value of biggest.
5.Stop.
```

SOURCE CODE:

```
x=int(input("Enter first number:"))
y=int(input("Enter second number:"))
z=int(input("Enter third number:"))
if x>=y and x>=z:
    biggest=x
elif y>=x and y>=z:
    biggest=y
else:
    biggest=z
print("The biggest number is", biggest)
```

OUTPUT:

```
Enter first number:234
Enter second number:567
Enter third number:123
The biggest number is 567
```

RESULT:

The program ran successfully and output is verified.

Experiment No:11 Lab Cycle:01 Date:23-10-2024

FILE EXTENSION

AIM:

Accept a file name from user and print extension of that.

ALGORITHM:

1. Input the file name from the user
2. Split the file name using the separator(.)
3. Extract the last part of the split result as file extension
4. Print the file extension

SOURCE CODE:

```
file_name = input("Enter the file name=")
file_extension = file_name.split('.')[-1]
print("The extension of the file is =", file_extension)
```

OUTPUT:

```
Enter the file name: example.txt
The extension of the file is = txt
```

RESULT

The program ran successfully and output is verified.

Experiment No:12 Lab Cycle:01 Date:06-11-2024

DISPLAY FIRST AND LAST COLOR

AIM:

Write a python program to create a list of colors from comma-separated color names entered by user. Display first and last colors.

ALGORITHM:

```
1.Start.
2. Input the Number of Colors: Prompt the user to enter count.
3. Initialize List: Create an empty list clrs.
4. Input Colors:
   Use a for loop to input count colors from the user.
   Append each color to clrs.
5. Display Results:
   Print the first color (clrs[0]) and the last color (clrs[-1]).
6. Stop.
```

SOURCE CODE:

```
clrs = []
count = int(input(“Enter the number of colors: “))

print(“Enter the colors:”)

 for x in range(count):
      color = input()
      clrs.append(color)

print(“First Color:”, clrs[0], “Last Color:”, clrs[-1])
```

OUTPUT:

```
Enter the number of colors: 3
Enter the colors: Red White Blue
Colors: Red White Blue
First Color: Red
Last Color: Blue
```

RESULT

The program ran successfully and output is verified.

Experiment No:13 Lab Cycle:01 Date:06-11-2024

COMPUTE n+nn+nnn

AIM:

Write a python program to accept an integer n and compute n+nn+nnn

ALGORITHM:

```
1. Start.
2. Input n: Read integer n.
3. Compute nn and nnn:
     Set nn = str(n) + str(n).
     Set nnn = str(n) + str(n) + str(n).
4. Calculate Result: Set res = int(n) + int(nn) + int(nnn).
5. Output: Print n + nn + nnn = res.
6. Stop.
```

SOURCE CODE:

```
n=(input(“ENTER THE VALUE OF ‘n’:”))

t1=n*2

t2=n*3

res (int(n)+int(t1)+int(t2))

print(n+”+”+t1+”+”+t2”=”+res)
```

OUTPUT:

```
ENTER THE VALUE OF ‘n’: 5

5+55+555=615
```

RESULT

The program ran successfully and output is verified.

Experiment No:14 Lab Cycle:01 Date:04-10-2024

PRINT COLORS FROM LIST

AIM:

Write a Python Program to print out all colors from color-list1 not contained in color-list2.

ALGORITHM:

1. Take space-separated input for List1 and store it in a set.
2. Take space-separated input for List2 and store it in a set.
3. Find the difference between List1 and List2.
4. Print the colors that are in List1 but not in List2.

SOURCE CODE:

```
list1 = set(input("Enter colors for List1 (separated by spaces): ").split())
list2 = set(input("Enter colors for List2 (separated by spaces): ").split())
diff = list1.difference(list2)
print("COLORS IN LIST1 NOT IN LIST2:", diff)
```

OUTPUT:

```
Enter the number of colors in List1: 3
Enter the colors to LIST1: Red Blue Green
Enter the number of colors in List2: 2
Enter the colors to LIST2: Red Green
COLORS IN LIST1 NOT IN LIST2: {'Blue'}
```

RESULT:

The program ran successfully and output is verified.

Experiment No:15 Lab Cycle:01 Date:10-10-2024

SWAPPING FIRST CHARACTERS

AIM:

Write a Python program to create single string separated with space from two strings by swapping the character at position 1.

ALGORITHM:

```
1. Input two strings, str1 and str2.
2. Swap their first characters:
     a)Replace the first character of str1 with the first character of str2.
     b)Replace the first character of str2 with the first character of str1.
3. Combine the modified strings with a space.
     a)Store the combined string in result.
4. Print the result.
```

SOURCE CODE:

```
str1 = input("Enter the first string: ")
str2 = input("Enter the second string: ")
swapped_str1=str2[0] + str1[1:]
swapped_str2=str1[0] + str2[1:]
result=swapped_str1+" "+swapped_str2
print("Swapped String: ",result)
```

OUTPUT:

```
Enter the first string: Hello
Enter the second string: World
Swapped String:  Wello Horld
```

RESULT:

The program ran successfully and output is verified.

Experiment No:16 Lab Cycle:01 Date:04-10-2024

SORTING DICTIONARY

AIM:

Write a Python Program to sort dictionary in ascending and descending order.

ALGORITHM:

1. Define a dictionary with key-value pairs.
2. Sort the dictionary in ascending order using sorted() and convert it to a dictionary using dict(). Store the result in asc.
3. Print the dictionary in ascending order.
4. Sort the dictionary in descending order using sorted() with reverse=True and convert it to a dictionary using dict(). Store the result in des.
5. Print the dictionary in descending order.

SOURCE CODE:

```
mydict = {'apple': 1, 'banana': 4, 'orange': 3, 'mango': 2}
asc = dict(sorted(mydict.items()))
print("Ascending order:", asc)
des = dict(sorted(mydict.items(), reverse=True))
print("Descending order:", des)
```

OUTPUT:

```
Ascending order: {'apple': 1, 'banana': 4, 'mango': 2, 'orange': 3}
Descending order: {'orange': 3, 'mango': 2, 'banana': 4, 'apple': 1}
```

RESULT:

The program ran successfully and output is verified.

Experiment No:17 Lab Cycle:01 Date:04-10-2024

MERGING 2 DICTIONARIES

AIM:

Write a Program to merge two dictionaries.

ALGORITHM:

1. Define the first dictionary(dict1)
2. Define the seconddictionary(dict2)
3. Use the update() function to merge dict2 into dict1
4. Print the merged dictionary

SOURCE CODE:

```
dict1 = {'a': 1, 'b': 2, 'c': 3, 'd': 4, 'e': 5}
dict2 = {'f': 6, 'g': 7}
dict1.update(dict2)
print("Merged Dictionary =", dict1)
```

OUTPUT:

```
Merged Dictionary = {'a': 1, 'b': 2, 'c': 3, 'd': 4, 'e': 5, 'f': 6, 'g': 7}
```

RESULT

The program ran successfully and output is verified.

Experiment No:18 Lab Cycle:01 Date : 09-09-2024

GCD

AIM:

Write a Python program to Find gcd of 2 numbers.

ALGORITHM:

```
1.Read two integers,num1 and num2.
2.If num1 is less than num2, swap the values of num1 and num2.
3.Repeat the following steps until num2 becomes zero:
    3.1.Compute the remainder of num1 divided by num2 and assign it to num2.
    3.2.Assign the value of num2 (before this operation) to num1.
4.When num2 becomes zero, the value of num1 is the GCD.
5.Print the GCD.
```

SOURCE CODE:

```
num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))
if num1 < num2:
    num1, num2 = num2, num1
while num2 != 0:
    num1, num2 = num2, num1 % num2

print(f"The GCD is: {num1}")
```

OUTPUT:

```
Enter the first number: 123
Enter the second number: 321
The GCD is: 3
```

RESULT:

The program ran successfully and output is verified.

Experiment No:19 Lab Cycle:01 Date:6-11-2024

CREATE A LIST REMAINING EVEN NUMBER

AIM:

From a list of integers ,create a list removing even numbers

ALGORITHM:

1. input a list of integers
2. check a number is divisible by 2 or not.
3. If number is divisible by 2, i.e, even number
4. Remove even munber from list
5. Print List

SOURCE CODE:

```
str_input = input("Enter the numbers, comma-separated: ")
listl [int(num) for num in str_input.split(',')]
print("List of numbers:", listl)
even=[ n for n in listl if n % 2 == 0]
print("Even numbers:", even)
```

OUTPUT:

```
How many elements: 3
Enter the element: 1
Enter the element: 3
Enter the element: 4
[1,3]
```

RESULT:

The program ran successfully and output is verified.

Experiment No:20 Lab Cycle:02 Date:10-10-2024

FACTORIAL OF N

AIM:

Write a Program to find the factorial of a number using a function.

ALGORITHM:

```
1. Define a function factorial(n) that:
2. a. Returns 1 if n is 0 or 1.
3. b. Calculates the factorial by multiplying numbers from 1 to n.
4. Take an integer n as input from the user.
5. Call the factorial() function with the input number.
6. Display the factorial result in the format "n! = result".
```

SOURCE CODE:

```
def factorial(n):
    if n == 0 or n == 1:
        return 1
    fact = 1
    for i in range(1, n + 1):
        fact *= i
    return fact
n = int(input("Enter a number: "))
result = factorial(n)
print(f"{n}! = {result}")
```

OUTPUT:

```
Enter a number: 5
5! = 120
```

RESULT:

The program ran successfully and output is verified.