Date: 10/11/2022

EXPERIMENT NO: 1
FIND LEAP YEAR OR NOT

AIM

Display future leap years from current year to a final year entered by user.

ALGORITHM

1. Read year and future year.
2. Iterate through list of years starting from current year to final year entered by user :
3. If given year is divisible by 4 or year is not divisible by 100 and year is divisible by 400 - Print year.

SOURCE CODE

```
currentYear = int(input("Enter the current year: "))
finalYear = int(input("Enter a future year: "))
print("Leap years in the giver period are:\n")
for Year in range(currentYear,futureYear+1):
   if ((Year % 400 == 0) or
         (Year % 100 != 0) and
         (Year % 4 == 0)):
      print(Year)
```

OUTPUT

```
Enter the current year: 2022
Enter final year: 2040
Leap years in the giver period are:
2024
2028
2032
2036
2040
```

RESULT

The program is executed successfully and result is obtained.

Date: 10/11/2022

EXPERIMENT NO: 2
FIND LIST OF VALUES

AIM

List comprehensions: (a) Generate positive list of numbers from a given list of integers (b) Square of N numbers (c) Form a list of vowels selected from a given word (d) List ordinal value of each element of a word.

(a) Generate positive list of numbers from a given list of integers

ALGORITHM

1. Create an empty list to store positive integers.
2. Iterate through the given list of integers:
3. If number greater than 0 then append the number to newly created positive integer list.
4. Print positive number list.

SOURCE CODE

```
int_list=[1,2,-1,5,-3,7,6,-7]
positiveList = []
for num in int_list:
   if(num>=0):
      positiveList.append(num)
print("Integer list: ", int_list)
print("Positive numbers from the list are: ", positiveList)
```

OUTPUT

Integer list: [1, 2, -1, 5, -3, 7, 6, -7]

Positive numbers from the list are: [1, 2, 5, 7, 6]

RESULT

The program is executed successfully and result is obtained.

(b) Square of N numbers

ALGORITHM

1. Request user input to find expected range of the list.
2. Create an empty list to store square of numbers.
3. Begin a loop starting from 1-range entered by user:
   For each iteration append value multiplied by itself [value x value]
4. Print the square list.

SOURCE CODE

```
n = int(input("Up to how many numbers do you want to calculate square of?"))
squareList=[]
for i in range(1,n+1):
    squareList.append(i*i)
print("Square list of numbers up to {}: ".format(n),squareList)
```

OUTPUT

```
Up to how many numbers do you want to calculate square of?5
Square list of numbers up to 5:  [1, 4, 9, 16, 25]
```

RESULT

The program is executed successfully and result is obtained.

(c) Form a list of vowels selected from a given word.

ALGORITHM

1. Create a list of vowels.
2. Request a user string input.
3. Iterate through each alphabet in the string:
   If alphabet if present in Vowel list:
   append alphabet to newly created list
4.Print newly created list

SOURCE CODE

```
vowels=["a","e","i","o","u"]
word = input("Enter a word")
vowelList = []
for alphabet in word:
    if alphabet in vowels:
        vowelList.append(alphabet)
print(vowelList)
```

OUTPUT

```
Enter a word: alphabet
['a', 'a', 'e']
```

RESULT

The program is executed successfully and result is obtained.

(d) List ordinal value of each element of a word.

ALGORITHM

1. Read a string input from user.
2. Iterate through each letter in the string
   Find ordinal value of each letter using ord() function : Append ordinal value of letter to list.
3. Print list.

SOURCE CODE

```
word = input("Enter a word: ")
ordList = []
for alphabet in word:
    ordList.append(ord(alphabet))
print(ordList)
```

OUTPUT

```
Enter a word: sample
[115, 97, 109, 112, 108, 101]
```

RESULT

The program is executed successfully and result is obtained.

Date: 10/11/2022

EXPERIMENT NO: 3

FIND THE OCCURRENCE OF WORD.

AIM

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

ALGORITHM

1. Read string input from user.
2. Split string into individual words and store in a dictionary.
3. Iterate through the dictionary elements and calculate count of each word by incrementing count value associated with that particular word.
4. Print the dictionary.

SOURCE CODE

```
sentence = input("Enter text to calculate occurance of each word: ")
counts = dict()
words = sentence.split()

for word in words:
    if word in counts:
        counts[word] += 1
    else:
        counts[word] = 1
print(counts)
```

OUTPUT

Enter text to calculate occurance of each word: sample word in a sample sentence to count the occurance of each each word

{'sample': 2, 'word': 2, 'in': 1, 'a': 1, 'sentence': 1, 'to': 1, 'count': 1, 'the': 1, 'occurance': 1, 'of': 1, 'each': 2}

RESULT

The program is executed successfully and result is obtained.

Date: 10/11/2022

EXPERIMENT NO: 4

FIND VALUES GREATER THAN 100 FROM A LIST

AIM

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

ALGORITHM

1. Read a list of integers from user.
2. For each input:
   If input value is greater than 100: Store "OVER" to the list
   else: Store input value to the list.
3. Print list

SOURCE CODE

```
len = int(input("How many numbers do you want to store? "))
inpValueList = []
for num in range(0,len):
    inpValue = int(input("Enter value"))
    if inpValue>100:  inpValueList.append("OVER")
    else: inpValueList.append(inpValue)
print(inpValueList)
```

OUTPUT

```
How many numbers do you want to store? 5
Enter value1
Enter value2
Enter value100
Enter value101
Enter value102
[1, 2, 100, 'OVER', 'OVER']
```

RESULT

The program is executed successfully and result is obtained.

Date: 10/11/2022

EXPERIMENT NO: 5

COUNT THE OCCURRENCE OF ‘a’

AIM

Store a list of first names. Count the occurrences of ‘a’ within the list.

ALGORITHM

1. Read n number of first names from user and store to list
2. Set count = 0
3. Iterate through list and add increment count by number of occurance of "a" in each word in list.
4. print count.

SOURCE CODE

```
firstNameList = input("Enter the first names seperated by whitespace: ").split()
count = 0
for name in firstNameList:
    count+=name.count('a')
print("Number of occurances of 'a' in this list: ",count)
```

OUTPUT

```
Enter the first names seperated by whitespace: adam john david joe peter
Number of occurances of 'a' in this list:  3
```

RESULT

The program is executed successfully and result is obtained.

Date: 05/12/2022

EXPERIMENT NO: 6
FIND THE LIST OF INTEGERS

AIM

Enter 2 lists of integers. Check (a) Whether list are of same length (b) whether list sums to same value (c) whether any value occur in both

ALGORITHM

1. Read 2 list of integers from user.
2. (a) If length of list 1 equal to length of list 2: then print "Two lists are of same length"
   Else: print "The lists are of different lengths".
3. (b) Set 2 variables sum1 and sum2 = 0.
4. Iterate through elements of list 1 and increment sum1 value by value of each element in list 1.
5. Iterate through elements of list 2 and increment sum2 value by value of each element in list 2.
6. If sum1 = sum2 then: print "The sum of values of elements in both lists are equal"
   else: print "The sum of values of elements in both lists are not equal"
7. (c) Iterate through elements of list 1:
   if element in list 2: append element to common[]
   print common[] list.

SOURCE CODE

```
fList = []
sList = []
common = []
sum1=0
sum2=0
len1 = int(input("How many numbers do you want to insert in first list?"))
for i in range(0,len1):
    inp = int(input())
    fList.append(inp)
len2 = int(input("How many numbers do you want to insert in second list?"))
for i in range(0,len2):
    inp = int(input())
    sList.append(inp)
if(len(fList) == len(sList)):
    print("Two lists are of same length")
```

```
else:
    print("Lists have different length")
for num in fList:
    sum1 += num
for num in sList:
    sum2 += num
if sum1==sum2:
    print("The sum of values of elements in both lists are equal")
else:
    print("Sum of values of both list's elements are different")
for num in fList:
    if num in sList:
        common.append(num)
        print("{} found in both lists".format(common))
```

OUTPUT

```
How many numbers do you want to insert in first list?3
1
2
3
How many numbers do you want to insert in second list?3
0
2
4
Two lists are of same length
The sum of values of elements in both lists are equal
[2] found in both lists
```

RESULT

The program is executed successfully and result is obtained.

Date: 05/12/2022

EXPERIMENT NO: 7

FIND THE OCCURRENCE OF FIRST CHARACTER OF STRING

AIM

Get a string from an input string where all occurrences of first character replaced with '$', except first character.

ALGORITHM

1. Read a string input from user (s).
2. Replace all occurance of first character (s[0]) of string (s) starting from second index of string (s1:)
3. print string (s)

SOURCE CODE

```
def changeOccurance(s):
    mod_string = s[1:].replace(s[0],"$")
    mod_string = s[0]+mod_string
    return mod_string
result = changeOccurance(input("Enter the string you want to modify: "))
print("Modified string : ",result)
```

OUTPUT

```
Enter the string you want to modify: onion
Modified string :  oni$n
```

RESULT

The program is executed successfully and result is obtained.

Date: 05/12/2022

EXPERIMENT NO: 8

CREATE A STRING FROM ANOTHER STRING

AIM

Create a string from given string where first and last characters exchanged.

ALGORITHM

1. Read an input string from user (str).
2. New string = first character (str[0]) + rest of the string excluding first and last character + last character (str[-1]).
3. Print New string.

SOURCE CODE

```
def replaceFirstWithLast(str):
    modifiedString = str[-1]+str[1:-1]+str[0]
    return modifiedString
result = replaceFirstWithLast(input("Enter the string you want to modify"))
print("Modified String = ", result)
```

OUTPUT

```
Enter the string you want to modify: python
Modified String =  nythop
```

RESULT

The program is executed successfully and result is obtained.

Date: 05/12/2022

EXPERIMENT NO: 9
FIND AREA OF CIRCLE

AIM

Accept the radius from user and find area of circle.

ALGORITHM

1. Read radius value from user.
2. Calculate area of circle using the formula pi*r**2 (pi=3.14)
3. Print calculated area.

SOURCE CODE

```
def findAreaOfCircle(r):
    area = 3.14 * r**2
    return area
r = int(input("Enter the radius of the circle"))
print("Area of the circle = ", findAreaOfCircle(r))
```

OUTPUT

```
Enter the radius of the circle10
Area of the circle =  314.0
```

RESULT

The program is executed successfully and result is obtained.

Date: 05/12/2022

EXPERIMENT NO: 10
FIND BIGGEST NUMBER

AIM

Find biggest of 3 numbers entered.

ALGORITHM

1. Read 3 number from user a,b,c respectively.
2. if a >b and a >c: print a
3. Else if b > c then: print b.
4. Else: print c.

SOURCE CODE

```
a= int(input("Enter the first number: "))
b= int(input("Enter the second number: "))
c= int(input("Enter the third number: "))
if a>b and a>c:
   print(a)
elif b>c:
   print(b)
else:
   print(c)
```

OUTPUT

```
Enter the first number: 3
Enter the second number: 4
Enter the third number: 10
10
```

RESULT

The program is executed successfully and result is obtained.

Date: 05/12/2022

EXPERIMENT NO: 11
FIND FILE EXTENSION

AIM

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

ALGORITHM

1. Read filename from user.
2. Split the filename into two parts using split function and with '.' as seperator.
3. Print extension.

SOURCE CODE

```
fn= input("Enter Filename: ")
f = fn.split(".")
print ("Extension of the file is : " + f[-1])
```

OUTPUT

```
Enter Filename: sample.py
Extension of the file is : py
```

RESULT

The program is executed successfully and result is obtained.

Date: 05/12/2022

EXPERIMENT NO: 12
FIND FIRST AND LAST COLORS

AIM

Create a list of colors from comma-separated color names entered by user. Display first and last colors.

ALGORITHM

1. Read list of colors from user and store in colorList.
2. Print first element in list (colorList[0]) and last element (colorList[-1]).

SOURCE CODE

```
def splitColorNames(str):
    c_list = str.split(",")
    return c_list
colors = input("Enter a series of color names seperated by comma")
colorList = splitColorNames(colors)
print("First color = ",colorList[0]," Last color = ",colorList[-1])
```

OUTPUT

```
Enter a series of color names seperated by comma: white,green,blue,yellow.black,orange
First color =  white  Last color =  orange
```

RESULT

The program is executed successfully and result is obtained.

Date: 05/12/2022

EXPERIMENT NO: 13

FIND AND COMPUTE n+nn+nnn

AIM

Accept an integer n and compute n+nn+nnn

ALGORITHM

1. Read integer input from user [number].
2. Result = number + (number * 10) + number + number * 100 + ((number * 10) + number)
3. Print result.

SOURCE CODE

```
num = int(input("Enter a number: "))
result = num + (num*10)+num + num*100+((num*10)+num)
print("Result = ", result)
```

OUTPUT

```
Enter a number: 3
Result =  369
```

RESULT

The program is executed successfully and result is obtained.

Date: 05/12/2022

EXPERIMENT NO: 14
FIND COLORS FROM A LIST

AIM

Print out all colors from color-list1 not contained in color-list2.

ALGORITHM

1. Read colorlist 1 and colorlist 2
2. Iterate through colors in colorlist 1:
   if color in colorlist 1 not present in colorlist 2: then print COLOR

SOURCE CODE

```
colorList1 = input("Enter color names for list 1 seperated by a comma").split(",")
colorList2 = input("Enter color names for list 2 seperated by a comma").split(",")
for color in colorList1:
    if color not in colorList2:
        print(color)
```

OUTPUT

```
Enter color names for list 1 seperated by a commawhite,green,orange,red,black
Enter color names for list 2 seperated by a commagreen,blue,yellow,magenta,purple
white
orange
red
black
```

RESULT

The program is executed successfully and result is obtained.

Date: 05/12/2022

EXPERIMENT NO: 15

CREATE TWO STRING AND SWAPPING CHARACTER

AIM

Create a single string separated with space from two strings by swapping the character at position 1.

ALGORITHM

1. Read string 1(str1) and string 2 (str2).
2. Set _str1 = first character of str2 (str2[0]) + characters of str1 excluding first character (str1[1:])
3. Set str2 = First character of str1 (st1[0]) + characters of str2 excluding first character (str2[1:])
4. Set newstring = _str1 + str2

SOURCE CODE

```
str1 = input("Enter the first string: ")
str2 = input("Enter the second string: ")
_str1 = str2[0] + str1[1:]
str2 = str1[0] + str2[1:]
newString = _str1 + " " + str2
print("Modified string : ", newString)
```

OUTPUT

```
Enter the first string: abc
Enter the second string: xyz
Modified string :  xbc ayz
```

RESULT

The program is executed successfully and result is obtained.

Date: 08/12/2022

EXPERIMENT NO: 16
SORT DICTIONARY

AIM

Sort dictionary in ascending and descending order.

ALGORITHM

1. Read a dictionary from user.
2. Sort the dictionary in ascending order by using sorted() function.
3. Print ascendingly sorted dictionary.
4. Sort the dictionary in descending order by using sorted() function and setting reverse argument 'True'.
5. Print descendingly sorted dictionary.

SOURCE CODE

```
dict = dict()
n = int(input("How many items do you want to enter to the dictionary?"))
for i in range(n):
    key = input("Enter the key: ")
    val = input("Enter value: ")
    dict[key] = val
print(dict)
print("Sorted [ascending]: ", sorted(dict.items()))
print("Sorted [descending]: ", sorted(dict.items(),reverse=True))
```

OUTPUT

How many items do you want to enter to the dictionary? 3

Enter the key: 1

Enter value: 10

Enter the key: 3

Enter value: 13

Enter the key: 2

Enter value: 9

{'1': '10', '3': '13', '2': '9'}

Sorted [ascending]: [('1', '10'), ('2', '9'), ('3', '13')]

Sorted [descending]: [('3', '13'), ('2', '9'), ('1', '10')]

RESULT

The program is executed successfully and result is obtained.

Date: 08/12/2022

EXPERIMENT NO: 17
MERGE DICTIONARIES

AIM

Merge two dictionaries.

ALGORITHM

1. Read 2 dictionaries d1,d2.
2. Merge d1 and d2 using update() function.
2. Print merge dictionary.

SOURCE CODE

```
def mergeDicts(x,y):
    x.update(y)
    return x
dictionary1 = {
    1:"a",2:"b",3:"c"
}
dictionary2 = {
    4:"d",5:"e",6:"f"
}
print(mergeDicts(dictionary1,dictionary2))
```

OUTPUT

```
{1: 'a', 2: 'b', 3: 'c', 4: 'd', 5: 'e', 6: 'f'}
```

RESULT

The program is executed successfully and result is obtained.

Date: 08/12/2022

EXPERIMENT NO: 18
FIND GCD

AIM

Find gcd of 2 numbers.

ALGORITHM

1. Read 2 numbers a,b from user.
2. Find factor list of a,b
3. Find highest common element from both lists.

SOURCE CODE

```
def gcd(x,y):
    if x ==0:
        return "GCD = {}".format(y)
    if y==0:
        return "GCD = {}".format(x)
    for i in range(1, min(x, y)):
        if x % i == 0 and y % i == 0:
            gcd = i
    return gcd
a = int(input("Enter the value of a: "))
b = int(input("Enter the value of b: "))
print("GCD: ",gcd(a,b))
```

OUTPUT

```
Enter the value of a: 32
Enter the value of b: 8
GCD:  4
```

RESULT

The program is executed successfully and result is obtained.

Date: 08/12/2022

EXPERIMENT NO: 19
CREATE A LIST REMOVING EVEN NUMBERS

AIM

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

ALGORITHM

1. Create a list of integers.
2. Iterate through list of integers: If number%2! = 0 then:
   Insert number to new list.
3. Print new list.

SOURCE CODE

```
_list = [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20]
__list=[]
for i in _list:
    if i%2!=0:
        __list.append(i)
print("Original list : {}".format(_list))
print("List after removing even numbers".format(__list))
```

OUTPUT

Original list : [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]

List after removing even numbers: [1, 3, 5, 7, 9, 11, 13, 15, 17, 19]

RESULT

The program is executed successfully and result is obtained.

Date: 12/12/2022

EXPERIMENT NO: 20

FACTORIAL

AIM

Program to find the factorial of a number.

ALGORITHM

Step 1: Start.
Step 2: Read integer input 'x' from user.
Step 3: Initialize factorial=1, i =1
Step 4: While i<=x repeat step 5 - 6:
Step 5: Set factorial = factorial * i
Step 6: set i = i+1
Step 7: Print factorial
Step 6: Stop.

SOURCE CODE

```
def fact(x):
    fact =1
    for i in range(1,x+1):
        fact = fact * i
        i+=1
    return fact
print("Factorial= ",fact(int(input("Enter a number: "))))
```

OUTPUT

Enter a number: 4

Factorial= 24

RESULT

The program is executed successfully and result is obtained.

Date: 12/12/2022

EXPERIMENT NO: 21
FIBONACCI SERIES

AIM

Generate Fibonacci series of N terms.

ALGORITHM

Step 1: Start.
Step 2: Read integer input 'x' from user.
Step 3: Initialize first =0, second = 1, third, i=0.
Step 4: Print first, second.
Step 5: While i<x-2 : repeat 7-9
Step 6: Set third = first+second.
Step 7: Print third
Step 8: Set first = second.
Step 9: Set second = third.
Step 10: Stop.

SOURCE CODE

```
def fibonacci(x):
    if(x<=0):
        print( "Cannot create a fibonacci sequence with this number")
        return
    first = 0
    second = 1
    print(first, end = " ")
    for i in range(x-1):
        print(second,end = " ")
        third = first + second
        first = second
        second = third
limit = fibonacci(int(input("Desired length of Fibonacci series: ")))
```

OUTPUT

Desired length of Fibonacci series: 7

0 1 1 2 3 5 8

RESULT

The program is executed successfully and result is obtained.

Date: 12/12/2022

EXPERIMENT NO: 22

SUM OF ITEMS IN A LIST

AIM

Find the sum of all items in a list.

ALGORITHM

Step 1: Start.
Step 2: Read integer input 'limit' from user.
Step 3: Initialize sum =0,i=0.
Step 4: While i< limit: repeat step 5-6
Step 5: Read value for list[i]
Step 6: Set i=i+1
Step 7: For each element in list: do Step 8
Step 8: sum = sum + element
Step 9: Print sum
Step 10: Stop

SOURCE CODE

```
limit = int(input("Enter the number of elements to insert into the list: "))
numList = []
for i in range(limit):
    val = int(input("Enter value: "))
    numList.append(val)
sumOfList = 0
for num in numList: sumOfList+=num
print("Sum of elements of list is {}".format(sumOfList))
```

OUTPUT

Enter the number of elements to insert into the list: 3

Enter value: 23

Enter value: 34

Enter value: 45

Sum of elements of list is 102

RESULT

The program is executed successfully and result is obtained.

Date: 12/12/2022

EXPERIMENT NO: 23

EVEN FOUR DIGIT PERFECT SQUARES

AIM

Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square

ALGORITHM

Step 1: Start.
Step 2: Read the lower and upper range for four-digit number.
Step 3: For each number in lower to upper range do: 4-9
Step 4: If number perfect square do: 5-9
Step 5: For each digit in number: do Step 6-9
Step 6: If digit is odd: do Step 7
Step 7: Return false.
Step 8: Else: do Step 9
Step 9: Print the number
Step 10: Stop.

SOURCE CODE

```
import math

def isPerfectSquare(x):
        if(x >= 0):
                root = int(math.sqrt(x))
                return ((root*root) == x)
        return False

excludeList = [1,3,5,7,9]
evenList = [2,4,6,8,0]
def allEven(num):
   while(num>0):
      if(num%10 in excludeList):
         return False
      else:
         num = num//10
   return True


def numberCombinations(x,y):
   for i in range(x,y+1):
      if (i//1000) not in excludeList:
         root = int(math.sqrt(i))
```

```
            if(isPerfectSquare(i)):
                if allEven(i):
                    print("{}".format(i))

def takeInput():
    s_range = int(input("Enter the starting range: "))
    f_range = int(input("Enter the stopping range: "))
    if s_range>= 1000 and f_range<10000:
        numberCombinations(s_range,f_range)
    else:
        print("Enter a range between 1000-9999")
        takeInput()

takeInput()
```

OUTPUT

```
Enter the starting range: 1000
Enter the stopping range: 9999
4624
6084
6400
8464
```

RESULT

The program is executed successfully and result is obtained.

Date: 14/12/2022

EXPERIMENT NO: 24

NUMBER PYRAMID

AIM

Display the given pyramid with step number accepted from user.

ALGORITHM

Step 1: Start
Step 2: Initialize i=1,x
Step 3: Read range from user
Step 4: While i<range: repeat Steps 5-9
Step 5: Set x=i
Step 6: While x<(i*i): repeat Step 7 and 8
Step 7: Print x
Step 8: x = x+i
Step 9: Print empty new line
Step 10: Stop

SOURCE CODE

```
def pyramid(r):
    for i in range(1,r+1):
        for x in range(i,i*i+1,i):
            print(x,end=" ")
        print("\n")

n = int(input("Enter a number: "))
pyramid(n)
```

OUTPUT

```
Enter a number: 4
1
2 4
3 6 9
4 8 12 16
```

RESULT

The program is executed successfully and result is obtained.

Date: 14/12/2022

EXPERIMENT NO: 25

CHARACTER FREQUENCY

AIM

Count the number of characters (character frequency) in a string.

ALGORITHM

Step 1: Start
Step 2: Create an empty dictionary wordCount
Step 3: Read a string input from user.
Step 4: For each alphabet in string: do
Step 5: Set wordCount[alphabet] = count(alphabet) in string
Step 6: Display wordCount dictionary.
Step 7: Stop

SOURCE CODE

```
def charFreq(str):
    wordCount = dict()
    for letter in set(str):
        wordCount[letter] = str.count(letter)
    return wordCount
word = input("Enter a string to check for character frequency: ")
print(charFreq(word))
```

OUTPUT

Enter a string to check for character frequency: sample

{'s': 1, 'a': 1, 'm': 1, 'p': 1, 'l': 1, 'e': 1}

RESULT

The program is executed successfully and result is obtained.

Date: 14/12/2022

EXPERIMENT NO: 26

STRING CONCATENATION

AIM

Add ‘ing’ at the end of a given string. If it already ends with ‘ing’, then add ‘ly’.

ALGORITHM

Step 1: Start
Step 2: Read string input from user.
Step 3: If last three characters of string = "ing" then do Step 4:
Step 4: Concatenate "ly" to string.
Step 5: Else do: Step 6:
Step 6: Concatenate "ing" to string.
Step 7: Display modified string to user.
Step 8: Stop

SOURCE CODE

```
def addIngOrLy(str):
    if(str[-3:] == "ing"):
        str= str + "ly"
    else:
        str = str + "ing"
    return str
word = input("Enter a word to modify: ")
modifiedString = addIngOrLy(word)
print("Modified string = ", modifiedString)
```

OUTPUT

Enter a word to modify: work

Modified string = working

RESULT

The program is executed successfully and result is obtained.

Date: 14/12/2022

EXPERIMENT NO: 27
LONGEST WORD

AIM

Accept a list of words and return length of longest word.

ALGORITHM

Step 1: Start
Step 2: Read string input from user.
Step 3: If last three characters of string = "ing" then do Step 4:
Step 4: Concatenate "ly" to string.
Step 5: Else do: Step 6:
Step 6: Concatenate "ing" to string.
Step 7: Display modified string to user.
Step 8: Stop

SOURCE CODE

```
def findLongestWord(wordList):
    highestLength =0
    for word in wordList:
        if(len(word)>highestLength):
            highestLength = len(word)
    return highestLength
words = input("Enter a series of words seperated by spaces: ").split(" ")
print("Longest word = ",findLongestWord(words),"characters long")
```

OUTPUT

Enter a series of words seperated by spaces: apple orange banana pineapple

Longest word = 9 characters long

RESULT

The program is executed successfully and result is obtained.

Date: 19/12/2022

EXPERIMENT NO: 28

PRINT PATTERN

AIM

Construct following pattern using nested loop.

```
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*
```

ALGORITHM

Step 1: Start
Step 2: Read desired pattern length from user.
Step 3: Initialize i=1,x=1
Step 4: While i<=limit : do repeat Step 5-8
Step 5: x=1
Step 6: While x<=i: do Step 7
Step 7: Print *
Step 8 Print empty new line.
Step 9: Stop

SOURCE CODE

```
def createPattern(limit):
    for i in range(1,limit+1):
        for x in range(i):
            print("*",end=" ")
        print("\n")
    for i in range(limit-1,0,-1):
        for x in range(i):
            print("*",end=" ")
        print("\n")
len = int(input("Max length of line: "))
createPattern(len)
```

OUTPUT

Max length of line: 5

```
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*
```

RESULT

The program is executed successfully and result is obtained.

Date: 19/12/2022

EXPERIMENT NO: 29

FACTORS LIST

AIM

Generate all factors of a number.

ALGORITHM

Step 1: Start
Step 2: Initialize i=1
Step 3: Read integer input 'x' from user to calculate factors.
Step 4: While i<=x do: 5-6
Step 5: If x modulo division i returns 0: do: Step 6
Step 6: Print i
Step 7: Stop

SOURCE CODE

```
def createFactorList(number):
    for i in range(1,number+1):
        if number%i == 0:
            print("{} \t".format(i),end=" ")
num = int(input("Enter a number to print it's factor list: "))
createFactorList(num)
```

OUTPUT

```
Enter a number to print it's factor list: 10
1       2       5       10
```

RESULT

The program is executed successfully and result is obtained.

Date: 19/12/2022

EXPERIMENT NO: 30
LAMBDA FUNCTIONS

AIM

Write lambda functions to find area of square, rectangle and triangle

ALGORITHM

Step 1: Start
Step 2: Read length of one side of square
Step 3: Return side*side
Step 4: Read length and breadth of rectangle
Step 5: Return length*breadth
Step 6: Read bread and height of triangle.
Step 7: Return 1/2 of breadth * height.
Step 8: Stop

SOURCE CODE

```
area_s=lambda a : a*a
area_rect=lambda l,b : l*b
area_triangle=lambda b1,h :0.5*b1*h
a=int(input("Enter the side of the square "))
print("Area of square ",area_s(a))
l=int(input("Enter the length of rectangle "))
b=int(input("Enter the breadth of rectangle "))
print("Area of rectangle ",area_rect(l,b))
b1=int(input("Enter the base of triangle "))
h=int(input("Enter the height of triangle "))
print("Area of triangle ",area_triangle(b1,h))
```

OUTPUT

Enter the side of the square 10

Area of square 100

Enter the length of rectangle 12

Enter the breadth of rectangle 12

Area of rectangle 144

Enter the base of triangle 10

Enter the height of triangle 5

Area of triangle 25.0

RESULT

The program is executed successfully and result is obtained.

Date: 02/01/23

EXPERIMENT NO: 31
BUILT -IN PACKAGES

AIM

Work with built-in packages.

ALGORITHM

Step 1: Start
Step 2: Import 3 packages ('math', 'random', 'datetime')
Step 3: The math package is used to find the square root of a number using the sqrt function.
Step 4:The random package is used to generate a random integer between 1 and 10 using the randint function.
Step 5: The datetime package is used to get the current date and time.
Step 6: Stop

SOURCE CODE

```
import math
import random
import datetime

num = 25
print("The square root of", num, "is", math.sqrt(num))
rand_num = random.randint(1, 10)
print("Random number between 1 and 10:", rand_num)
now = datetime.datetime.now()
print("Current date and time:", now)
formatted_time = now.strftime("%Y-%m-%d %H:%M:%S")
print("Formatted date and time:", formatted_time)
```

OUTPUT

The square root of 25 is 5.0

Random number between 1 and 10: 10

Current date and time: 2023-02-16 07:32:37.454551

Formatted date and time: 2023-02-16 07:32:37

RESULT

The program is executed successfully and result is obtained.

Date: 02/01/23

EXPERIMENT NO: 32
GRAPHICS PACKAGE

AIM

Create a package with modules rectangle, circle and sub-package 3D-graphice with modules cuboid and sphere. Include methods to find the area and perimeter of respective figures in each module. Write programs that finds area and perimeter of figures by different importing statements. (Include selective import of modules and import * statements).

ALGORITHM

Step 1: Create a new directory named "graphics" in your desired location.

Step 2: Inside the "graphics" directory, create directory named "3D".

Step 3: Inside graphics directory create two Python files named "rectangle.py" and "circle.py".

Step 4: Inside the "3D" directory, create two Python files named "cuboid.py" and "sphere.py".

Step 5: Create methods for calculating the area and perimeter of each shape in their respective Python files.

Step 6: Create an init.py file in both the "graphics" and "3D" directories to turn them into packages.

Step 7: Test the packages by importing them to a new Python file using appropriate methods.

SOURCE CODE

```
from graphics.rectangle import *
from graphics.circle import *
from graphics.graphics3D import cuboid
from graphics.graphics3D import sphere

def operations():
    ch = int(input("\nSelect a shape to calculate it's Area and Perimeter/Circumference?\n0. Exit
1.Rectangle 2.Circle 3.Cuboid 4.Sphere\n"))
    if ch == 1:
        length = int(input("Length? "))
        breadth = int(input("Breadth? "))
        print("Area = {:.2f}".format(rectangle_area(length, breadth)))
        print("Perimeter = {:.2f}".format(rectangle_perimeter(length,breadth)))
    elif ch == 2:
        radius = int(input("Radius? "))
        print("Area = {:.2f}".format(circle_area(radius)))
        print("Perimeter = {:.2f}".format(circle_circumference(radius)))
```

```
    elif ch == 3:
        length = int(input("Length? "))
        breadth = int(input("Breadth? "))
        height = int(input("Height? "))
        print("Area = {:.2f}".format(cuboid.cuboid_surface_area(length,breadth,height)))
        print("Perimeter = {:.2f}".format(cuboid.cuboid_perimeter(length,breadth,height)))
    elif ch == 4:
        radius = int(input("Radius? "))
        print("Area = {:.2f}".format(sphere.sphere_surface_area(radius)))
        print("Perimeter = {:.2f}".format(sphere.sphere_circumference(radius)))
    elif ch == 0:
        return
    operations()

operations()
```

PACKAGE (GRAPHICS):

rectangle.py

```
def rectangle_area(length, width):
    return length * width
def rectangle_perimeter(length, width):
    return 2 * (length + width)
```

circle.py

```
import math
def circle_area(radius):
    return math.pi * (radius ** 2)
def circle_circumference(radius):
    return 2 * math.pi * radius
```

SUBPACKAGE (GRAPHICS3D):

cuboid.py

```
def cuboid_surface_area(length, width, height):
    return 2 * (length * width + width * height + height * length)
def cuboid_perimeter(length, width, height):
    return 4* (length + width + height)
```

sphere.py

```
import math

def sphere_surface_area(radius):
    return 4 * math.pi * (radius ** 2)
def sphere_circumference(radius):
    return 2 * math.pi * radius
```

OUTPUT

Select a shape to calculate it's Area and Perimeter/Circumference?

0. Exit 1.Rectangle 2.Circle 3.Cuboid 4.Sphere: 1

Length? 10

Breadth? 10

Area = 100.00

Perimeter = 40.00

Select a shape to calculate it's Area and Perimeter/Circumference?

0. Exit 1.Rectangle 2.Circle 3.Cuboid 4.Sphere: 2

Radius? 10

Area = 314.16

Perimeter = 62.83

Select a shape to calculate it's Area and Perimeter/Circumference?

0. Exit 1.Rectangle 2.Circle 3.Cuboid 4.Sphere: 3

Length? 10

Breadth? 10

Height? 10

Area = 600.00

Perimeter = 120.00

Select a shape to calculate it's Area and Perimeter/Circumference?

0. Exit 1.Rectangle 2.Circle 3.Cuboid 4.Sphere: 4

Radius? 10

Area = 1256.64

Perimeter = 62.83

Select a shape to calculate it's Area and Perimeter/Circumference?

0. Exit 1.Rectangle 2.Circle 3.Cuboid 4.Sphere: 0

RESULT

The program is executed successfully and result is obtained.

Date: 02/01/23

EXPERIMENT NO: 33

RECTANGLE CLASS

AIM

Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare two rectangle objects by their area.

ALGORITHM

Step 1: Start

Step 2: Create a Rectangle class that has length and breadth attributes and methods to calculate and display its area and perimeter.

Step 3: Read the Length and Breadth of the 2 Rectangles from user

Step 4: Instantiate 2 rectangles with the user inputs as its arguments.

Step 5: Compare the area of both rectangles using its area method and display which is larger.

Step 6: Stop

SOURCE CODE

```
class Rectangle:
    def __init__(self, length, breadth):
        self.length = length
        self.breadth = breadth

    def area(self):
        return self.length * self.breadth

    def perimeter(self):
        return 2 * (self.length + self.breadth)

length1 = int(input("Enter the length of First rectangle: "))
breadth1 = int(input("Enter the breadth of First rectangle: "))
r1 = Rectangle(length1,breadth1)
length2 = int(input("Enter the length of Second rectangle: "))
breadth2 = int(input("Enter the breadth of Second rectangle: "))
r2 = Rectangle(length2,breadth2)
print("Rectangle 1 area:", r1.area())
print("Rectangle 2 area:", r2.area())
print("Rectangle 1 perimeter:", r1.perimeter())
```

```
print("Rectangle 2 perimeter:", r2.perimeter())

if r1.area()>r2.area():
   print("\nRectangle 1 is bigger")
elif r1.area() == r2.area():
   print("\nBoth rectangles are equal in size")
else:
   print("\nRectangle 2 is bigger")
```

OUTPUT

Enter the length of First rectangle: 10

Enter the breadth of First rectangle: 20

Enter the length of Second rectangle: 30

Enter the breadth of Second rectangle: 40

Rectangle 1 area: 200

Rectangle 2 area: 1200

Rectangle 1 perimeter: 60

Rectangle 2 perimeter: 140

Rectangle 2 is bigger

RESULT

The program is executed successfully and result is obtained.

Date: 02/01/23

EXPERIMENT NO: 34

BANK ACCOUNT

AIM

Create a bank account with members account number, name, type of account and balance. Write constructor and methods to deposit at the bank and withdraw an amount from the bank.

ALGORITHM

Step 1: Start

Step 2: Create a BankAccount class with 'account no', 'account holder name', 'account type', 'account balance' attributes and methods to Create an account, deposit and withdraw money.

Step 3: Read user input to select an operation.

Step 4: If choice is 1 : Read 'account holder name', 'account number', 'account type' from user and set initial 'account balance' to 0

Step 5: If choice is 2: Read 'amount'. Increment balance by 'amount'.

Step 6: If choice is 3: Read 'amount'. If 'account balance' > = 'amount' : Decrement 'account balance' by 'amount' and display current 'account balance'. Else display error message.

Step 7: Stop

SOURCE CODE

```
class BankAccount:
    def __init__(self, account_number, name, account_type, balance=0):
        self.account_number = account_number
        self.name = name
        self.account_type = account_type
        self.balance = balance
        self.accountInfo();

    def accountInfo(self):
        print("**Account Information**\n")
        print(f"Name: {self.name}")
        print(f"Account Number: {self.account_number}")
        print(f"Account Type: {self.account_type}")
        print(f"Balance: {self.balance}")
    def deposit(self, amount):
        self.balance += amount
```

```
        print(f"Deposited {amount:.2f}. Current balance is {self.balance:.2f}.")

    def withdraw(self, amount):
        if self.balance < amount:
            print("Insufficient balance.")
        else:
            self.balance -= amount
            print(f"Withdrew {amount:.2f}. Current balance is {self.balance:.2f}.")


while True:
    ch = int(input(
        "\nEnter the operation you want to perform: \n0. Exit 1. Create a new account 2.Deposit
money 3. Withdraw money\n\n"))
    if ch == 0:
        break
    elif ch == 1:
        name = input("Enter your name: ")
        acNo = input("Enter an account number: ")
        acType = input("Enter your desired account type: ")
        acBalance = 0
        newAccount = BankAccount(acNo, name, acType, acBalance)
    elif ch == 2:
        amount = int(input("Amount: "))
        newAccount.deposit(amount)
    elif ch == 3:
        amount = int(input("Amount: "))
        newAccount.withdraw(amount)
```

OUTPUT

Enter the operation you want to perform:

0. Exit 1. Create a new account 2.Deposit money 3. Withdraw money: 1

Enter your name: John Doe

Enter an account number: 1234

Enter your desired account type: Savings

Account Information

Name: John Doe

Account Number: 1234

Account Type: Savings

Balance: 0

Enter the operation you want to perform:

0. Exit 1. Create a new account 2.Deposit money 3. Withdraw money: 2

Amount: 100

Deposited 100.00. Current balance is 100.00.

Enter the operation you want to perform:

0. Exit 1. Create a new account 2.Deposit money 3. Withdraw money: 3

Amount: 10

Withdrew 10.00. Current balance is 90.00.

Enter the operation you want to perform:

0. Exit 1. Create a new account 2.Deposit money 3. Withdraw money: 0

RESULT

The program is executed successfully and result is obtained.

Date: 09/01/23

EXPERIMENT NO: 35

RECTANGLE CLASS WITH PRIVATE ATTRIBUTE

AIM

Create a class Rectangle with private attributes length and width. Overload '<' operator to compare the area of 2 rectangles.

ALGORITHM

Step 1: Start

Step 2: Define a class 'Rectangle' with the following private attributes: length, breadth.

Step 3: Define a method named area that calculates and returns the area of the rectangle as the product of its length and width.

Step 4: Define the < operator by overloading the __lt__ method. This method takes another Rectangle object other as an argument, calculates the area of the current rectangle using the area method, calculates the area of the other rectangle by calling its area method, and compares the two areas using the < operator. It then returns the result of the comparison.

Step 5: Create two Rectangle objects, rect1 and rect2, with their respective length and width values.

Step 6: Compare the areas of the two rectangles using the < operator. If rect1 has a smaller area than rect2, print a message indicating that. Otherwise, print a message indicating that rect1 has a larger area than rect2.

Step 7: Stop

SOURCE CODE

```
class Rectangle:
def __init__(self, length, breadth):
self.length = length
self.breadth = breadth

def area(self):
return self.length * self.breadth

def perimeter(self):
return 2 * (self.length + self.breadth)

length1 = int(input("Enter the length of First rectangle: "))
breadth1 = int(input("Enter the breadth of First rectangle: "))
r1 = Rectangle(length1,breadth1)
```

```
length2 = int(input("Enter the length of Second rectangle: "))
breadth2 = int(input("Enter the breadth of Second rectangle: "))
r2 = Rectangle(length2,breadth2)
print("Rectangle 1 area:", r1.area())
print("Rectangle 2 area:", r2.area())
print("Rectangle 1 perimeter:", r1.perimeter())
print("Rectangle 2 perimeter:", r2.perimeter())

if r1.area()>r2.area():
print("\nRectangle 1 is bigger")
elif r1.area() == r2.area():
print("\nBoth rectangles are equal in size")
else:
print("\nRectangle 2 is bigger")
```

OUTPUT

Enter the length of First rectangle: 10

Enter the breadth of First rectangle: 20

Enter the length of Second rectangle: 10

Enter the breadth of Second rectangle: 15

Rectangle 1 has a larger area than Rectangle 2

RESULT

The program is executed successfully and result is obtained.

Date: 09/01/23

EXPERIMENT NO: 36

OVERLOADING '+' OPERATOR

AIM

Create a class Time with private attributes hour, minute and second. Overload '+' operator to find the sum of 2 time.

ALGORITHM

Step 1: Start

Step 2: Define a class named Time with the following attributes: hour, minute, second.

Step 3: Define a constructor method __init__ that takes the three attributes (hour, minute, and second) as arguments and assigns them to the private instance variables hour, minute, and second.

Step 4: Define the + operator by overloading the __add__ method. This method takes another Time object other as an argument, calculates the total number of seconds in both time objects, adds them together, and creates a new Time object with the result.

'hour' is calculated using the formula: total_seconds // 3600,
'minute': (total_seconds // 60) % 60
'second': total_seconds % 60

Step 5: Display the new Time.

Step 6: Stop

SOURCE CODE

```
class Time:
    def __init__(self, hour, minute, second):
        self.hour = hour
        self.minute = minute
        self.second = second

    def __add__(self, other):
        total_seconds = self.hour * 3600 + self.minute * 60 + self.second
        total_seconds += other.hour * 3600 + other.minute * 60 + other.second
        return Time(total_seconds // 3600, (total_seconds // 60) % 60, total_seconds % 60)
print("Enter first time [H:M:S]\n")
h1 = int(input("Hour:"))
m1 = int(input("Minute:"))
s1 = int(input("Second:"))
time1 = Time(h1, m1, s1)
```

```
print("\nEnter second time [H:M:S]\n")
h2 = int(input("Hour:"))
m2 = int(input("Minute:"))
s2 = int(input("Second:"))
time2 = Time(h2, m2, s2)

sum_time = time1 + time2
print(sum_time.hour, sum_time.minute, sum_time.second)
```

OUTPUT

Enter first time [H:M:S]

Hour:2

Minute:40

Second:10

Enter second time [H:M:S]

Hour:4

Minute:20

Second:50

7:1:0

RESULT

The program is executed successfully and result is obtained.

Date: 09/01/23

EXPERIMENT NO: 37

BOOK CLASS

AIM

Create a class Publisher(name) .Derive class Book from Publisher with attributes title and author. Derive class Python from Book from Publisher with attributes price and no_of_pages. Write a program that displays information about a Python book. Use base class constructor invocation and method overloading.

ALGORITHM

Step 1: Start

Step 2: Define a class named Publisher with a single attribute name.

Step 3: Define a constructor method __init__ for the Publisher class that takes a single argument name and assigns it to the name attribute.

Step 4: Define a class named Book that derives from the Publisher class with two attributes title and author.

Step 5: Define a constructor method __init__ for the Book class that takes three arguments name, title, and author, calls the constructor of the base class (Publisher) with the name argument using super().__init__(name), and assigns the title and author arguments to the respective attributes.

Step 6: Define a method display for the Book class that prints the title and author of the book.

Step 7: Define a class named Python that derives from the Book class with two attributes price and num_pages.

Step 8: Define a constructor method __init__ for the Python class that takes five arguments name, title, author, price, and num_pages, calls the constructor of the base class (Book) with the name, title, and author arguments using super().__init__(name, title, author), and assigns the price and num_pages arguments to the respective attributes.

Step 9: Create an instance of Python class with custom info and use display method to display the properties.

Step 10: Stop

SOURCE CODE

```
class Publisher:
    def __init__(self, name):
        self.name = name

class Book(Publisher):
    def __init__(self, name, title, author):
        super().__init__(name)
        self.title = title
        self.author = author

    def display(self):
        print("Title:", self.title)
        print("Author:", self.author)

class Python(Book):
    def __init__(self, name, title, author, price, num_pages):
        super().__init__(name, title, author)
        self.price = price
        self.num_pages = num_pages

    def display(self):
        super().display()
        print(f"Price:Rs.{self.price}")
        print("Number of Pages:", self.num_pages)


python_book = Python("Createspace Independent Pub ", "Python Programming", "Ramsey
Hamilton", 2700, 90)
python_book.display()
```

OUTPUT

Title: Python Programming

Author: Ramsey Hamilton

Price:Rs.2700

Number of Pages: 90

RESULT

The program is executed successfully and result is obtained.

Date: 09/01/23

EXPERIMENT NO: 38

READ LINE BY LINE

AIM

Write a Python program to read a file line by line and store it into a list.

ALGORITHM

Step 1: Start

Step 2: Check if the specified file exists. Else create a new file with few lines of data.

Step 3: Open file in read mode and read each line individually.

Step 4: Append read lines to a list.

Step 5: Print list.

Step 6: Stop

SOURCE CODE

```
from os.path import exists

filename = "MyFile.txt"

def LinesToList():
    file = open(filename,"r")
    lines = file.readlines()
    return lines
if not exists(filename):
    with open(filename,"w") as file:
        for i in range(1, 5):
            file.write(f"Sample line{i}\n")

print(f"List: {LinesToList()}")
```

OUTPUT

List: ['Sample line1\n', 'Sample line2\n', 'Sample line3\n', 'Sample line4\n']

RESULT

The program is executed successfully and result is obtained.

Date: 16/01/23

EXPERIMENT NO: 39

COPY ODD LINES FROM FILE

AIM

Python program to copy odd lines of one file to other

ALGORITHM

Step 1: Start

Step 2: If sourceFile doesn't exist create one with few lines of data.

Step 3: Open sourceFile in read mode and copy all lines to a list.

Step 4: Initialize 'count' as 0

Step 5: Open destinationFile in write mode:

Step 6: Iterate through list and increment count with each iteration.

Step 7: For each iteration where count%2 is not 0 write line to destinationFile.

Step 8: Print content of both files.

Step 9: Stop

SOURCE CODE

```
from os.path import exists
sourceFileName = "file1"
destinationFileName = "file2"

def readOddLines():
   with open(sourceFileName) as sourceFile:
      with open(destinationFileName, "w") as destinationFile:
         count = 0
         linesList = sourceFile.readlines()
         for line in linesList:
            count = count+1
            if count%2 != 0:
               destinationFile.write(line)
print("Contents of sourceFile:")
if not exists(sourceFileName):
   with open(sourceFileName,"w") as sourceFile:
      for i in range(1,6):
```

```
            sourceFile.write(f"Line {i}")
readOddLines()
with open(sourceFileName) as file:
    print(file.read())
print("Contents of Destination file:")
with open(destinationFileName) as file:
    print(file.read())
```

OUTPUT

Contents of sourceFile:

Line 1

Line 2

Line 3

Line 4

Line 5

Contents of Destination file:

Line 1

Line 3

Line 5

RESULT

The program is executed successfully and result is obtained.

Date: 16/01/23

EXPERIMENT NO: 40

CSV READ ROWS

AIM

Write a Python program to read each row from a given csv file and print a list of strings

ALGORITHM

Start 1:Start

Step 2: Import the csv module.

Step 3: Open the file "fakeData.csv" and store it in a variable.

Step 4: Use the csv.reader() function to read the data in csvfile.

Step 5: Iterate over each row in data.

Step 6: For each row. append data to a list called rows.

Step 7: Iterate through rows list and print each row.

Step 8:Stop

SOURCE CODE

```
import csv

def read_csv_file(file_path):
    with open(file_path, 'r') as file:
        csv_reader = csv.reader(file)
        rows = []
        for row in csv_reader:
            rows.append(row)
        return rows


file_path = 'fakeData.csv'
rows = read_csv_file(file_path)
for row in rows:
    print(row)
```

OUTPUT

['S.No', 'First Name', 'Last Name', 'Salary']

['1', 'John', 'Doe', '50000']

['2', 'Jane', 'Smith', '65000']

['3', 'Mark', 'Johnson', '75000']

['4', 'Sara', 'Williams', '60000']

['5', 'David', 'Miller', '80000S']

RESULT

The program is executed successfully and result is obtained.

Date: 16/01/23

EXPERIMENT NO: 41

CSV READ SPECIFIC ROWS

AIM

Write a Python program to read specific columns of a given CSV file and print the content of the columns.

ALGORITHM

Start 1:Start

Step 2: Import the csv module.

Step 3: Read the columns that user want to display specifically to a list 'columns'.

Step 4: Open the file "fakeData.csv" and store it in a variable.

Step 5: Use the csv.reader() function to read the data in csvfile.

Step 6: Iterate over each row in data.

Step 7: For each row:
Iterate through list columns:
Append row[column] to new list:

Step 8: Return list

Step 9: Iterate through list and print each row.

Step 10: Stop

SOURCE CODE

```
import csv

def read_csv_columns(file_path, columns):
    with open(file_path, 'r') as file:
        csv_reader = csv.reader(file)
        rows = []
        for row in csv_reader:
            selected_cols = [row[i] for i in columns]
            rows.append(selected_cols)
        return rows
def selectColumns():
```

```
    inputs = input("Please specify the columns you wish to include starting from 1-4 \n[Use space to
seperate each number]: ").split()
    columns = []
    for column in inputs:
        columns.append(int(column)-1)
    return columns

file_path = 'fakeData.csv'
columns = selectColumns()
rows = read_csv_columns(file_path, columns)
for row in rows:
    print(row)
```

Input CSV file (fakeData.csv):

S.No,First Name,Last Name,Salary

1,John,Doe,50000

2,Jane,Smith,65000

3,Mark,Johnson,75000

4,Sara,Williams,60000

5,David,Miller,80000S

OUTPUT

Please specify the columns you wish to include starting from 1-4

[Use space to seperate each number]: 1 4

['S.No', 'Salary']

['1', '50000']

['2', '65000']

['3', '75000']

['4', '60000']

['5', '80000S']

RESULT

The program is executed successfully and result is obtained.

Date: 16/01/23

EXPERIMENT NO: 42

DICTIONARY TO CSV FILE

AIM

Write a Python program to write a Python dictionary to a csv file. After writing the CSV file read the CSV file and display the content.

ALGORITHM

Start 1:Start

Step 2: Import the csv module.

Step 3: Open the file "FakeData_.csv" in write mode.

Step 4: Insert dictionary.keys() as first row for the csv file.

Step 5: Insert dictionary.values() as next row for the csv file.

Step 6: Open the file in read mode.

Step 7: Use the csv.reader() function to read the data in csvfile.

Step 8: Append each row in csvfile to new List.

Step 9: Return List

Step 10: Print List

Step 11:Stop

SOURCE CODE

```
import csv

def write_dict_to_csv(file_path, data):
    with open(file_path, 'w', newline='') as file:
        writer = csv.writer(file)
        writer.writerow(data.keys())
        writer.writerow(data.values())

def read_csv_file(file_path):
    with open(file_path, 'r') as file:
        csv_reader = csv.reader(file)
        rows = []
```

```
        for row in csv_reader:
            rows.append(row)
        return rows


file_path = 'FakeData_.csv'
data = {'name': 'John Doe', 'age': 35, 'city': 'New York'}
write_dict_to_csv(file_path, data)
rows = read_csv_file(file_path)
for row in rows:
    print(row)
```

OUTPUT

['Name', 'Age', 'City']

['John Doe', '35', 'New York']

RESULT

The program is executed successfully and result is obtained.