Sri Venkateswara College of Engineering & Technology

Autonomous

Department of Computer Science

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Aim:- Installation and configuration of python in Windows.

Description:-

* point your web browser to the doconload page on the python website.
* Select the latest windows X86 MSI or version 3.8.1 and click on it and download
* And make sure that whether it is download or not
* Run the installer
* "Select" install for all users and click the next>button
* Keep the default options as the destination directory and click next> button again
* Dont make any change in the "customize Python 3.8" dialog. just click Next>
* Click yes if asked if the program should be allowed to install. Software on your System
* Click the ~~install~~ finish button when installation complets.

Running:- Installing process creates a python 3.8.1 section under start menu-all program

Result: Thus I have successfully installed Python for windows and run it successfully.

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Aim: To develop a python program to compute the GCD of two numbers.

Algorithm:-

1. Start
2. Define and declare the function for GCD
2.1. If x is less than y, then x is small
2.2. If y is less than x, then y is small
2.3. for i in range 1 and small +1 do
   2.3.1 - If x mod i is 0 and
   y mod i is 0 and then gcd is i
3. Input the first number from user
4. Input the second number from user
5. print the GCD of 2 number
6. Stop.

program:-

```
num1 = float (input ("enter the first value :"))
num2 = float (input ("enter the second value:"))
a = num1
b = num2
while (num2!=0):
  temp = num2
  num2 = num1 %. num2
  num1 = temp
  gcd = num1
```

Output:

Enter the first value: 10

Enter the second value: 30

GCD of 10.0 and 30.0 = 10.0

Process finished.

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

```
print ("\n GCD of {0} and {1} = {2}", format (a, b, gcd))
```

Result: Thus, the GCD of two numbers is executed Succes sfully and results is observed carefully.

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Aim:- To develop a python program to find the Square root of a given number. (Newtons Method)

Algorithm :-

1. Start
2. Define the square root function
3. Assign n value to the x value
4. Now, start a loop and keep calculating the root which will surely move towards the correct square root of n
5. Check for the difference between the assumed 'x' and calculated root, if not yet inside tolerance then update root and continue
6. If the calculated root comes inside the tolerance allowed then break out of the loop.
7. print the root
8. Stop.

program:-

```
def newton_method (number, number_iters=100):
    a = float (number)
    for i in range (number_iters):
        number = 0.5 * (number + a/number)
        return number
a = int(input("Enter first number:"))
```

Output:

```
Enter the first number : 20
Enter the second number: 30
Squareroot of first number : 4.472
Squareroot of second number : 5.477
process finished.
```

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

```
b = int(input(" Enter second number:"))
print("Squareroot of first number:", newton_method(a))
print("Squareroot of Second number:", newton_method(b))
```

Result:

Thus, the Squareroot of a number is executed successfully and output is noted carefully.

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Aim:- To develop a python program to implement linear search functions.

Algorithm :-

1. start
2. Define function/ method for linear search and pass arr, x to the method
3. Start for i loop in range of length of array.
4. Check if condition of arr[i] equal to x then return i value
5. otherwise -1
6. Enter the elements in the array and x value
7. print as element found at index
8. Stop

program:-

```
def linear_search(alist, key):
    for i in range(len(alist)):
        if alist[i] == key:
            return:
        return -1
alist = input("enter the list:")
alist = alist.split()
alist = [int(x) for x in a list]
```

Output:

Enter the list : 10 20 30 40 50

Enter the value to search : 40

40 was found at index 3.

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

```
key = int(input("Enter a value to search:"))
index = linear_search(alist, key)
if index<0:
    print('{} was not found'.format(key))
else
    print('{} was found at index {}'.format(key, index))
```

Result:
The Python program for linear search is successfully executed and output is observed carefully.

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Aim: To develop a python program to implement the binary search using functions.

Algorithm:-

1. Start
2. Define method/function for binary search and pass parameters.
3. Declare and initialize the variables
4. Start while loop as low less than equal to high then calculate mid
5. If mid less than element then low equal to mid plus 1
6. Else if mid greater than element then ~~low~~ high equal to mid-1
7. Otherwise return mid
8. (or) Otherwise return-1
9. Enter the elements into the list
10. print the result with index
11. Stop.

program:-

```
def binary_search (alist, key):
    start = 0
    end = len(alist)
    while start < end:
        mid = (start+end)//2
        if alist[mid] > key:
```

Output:

Enter the sorted list: 10 12 14 16 34 38 45 56 78 89 98

The No. to search for: 78

78 was found at index 8.

process finished.

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

```
    end = mid
  elif alist[mid]<key:
      start = mid+1
    else:
      return mid
      return -1
alist = input('Enter the sorted list:')
alist = alist.split()
alist = [int(x) for x in alist]
key = int(input('The no. to seach for:'))
index = binary_search(alist, key)
if index<0:
   print('{} was not found'.format(key))
else:
   print('{} was found at index {}'.format(key,index))
```

Result:-

Thus, the python program for binary search is successfully executed and result is noted carefully.

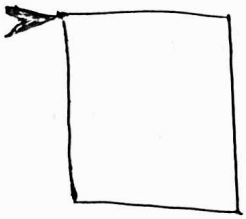
Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Aim: To develop a Python program that illustrates turtle module

Algorithm:-

1. Start
2. We have to import the turtle module
3. Enter the background color and pensize
4. Define the cure function
5. Start the loop upto range and gave right and forward length
6. Enter the penspeed outside color and inside colour of heart
7. Enter to fill the heart and enter to draw the curve from left to right.
8. At last we have to end the fill
9. Stop.

Output :



Program No. ☐☐ Date : ☐☐-☐☐-☐☐

program:

```
#import turtle library
import turtle
my_pen = turtle.Turtle()
for i in range[4]:
    my_pen.forward(50)
    my_pen.right(90)
turtle.done()
```

Result:

Thus, the program that implements the turtle mode is executed successfully.

Program No. [ ][ ] Date : [ ][ ]-[ ][ ]-[ ][ ]

Aim: Write a python program to develop a insertion sort.

Algorithm:-

1. Start
2. Create a function insertion sort that takes a list as argument.
3. Inside the function create a loop with a loop variable i that counts from 1 to the length of the list -1
4. Set temp equal to the element of index 'i'
5. Set 'j' equal to i-1;
6. Creat a while loop that runs as long as 'j' is non-negative and 'temp' is Smaller than the element at index 'j' ~~and~~
7. Inside the while loop, Set the element at index 'j+1' equal to the element at index 'j' and decrement 'j'.
8. After the while loop finishes. Set the element at index 'j+1' equal to temp.
9. Stop.

Output:

Enter the list: 90 81 81 45 63 92

Sorted list : [21 34 45, 63, 92, 90]

Process finished.

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

program:-

```
def insertion_sort(alist):
    for i in range(1, len(alist)):
        temp = alist[i]
        j = i-1
        while(j>=0 and temp<alist[j]):
            alist[j+1] = alist[j]
            j = j-1
            alist[j+1] = temp
alist = input("Enter the list:")
alist = alist.split()
alist = [int(x) for x in alist]
insertion_sort(alist)
print("Sorted list:", end = ")
print(alist)
```

Result: Thus, the python program to implement of insertion Sort is successfully executed.

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Aim: To develop a Python program to implement the selection sort.

Algorithm :-

1. Start
2. Create a function Selection.sort that takes a list as argument.
3. Inside the function create a loop with a loop variable 'i' that counts from 0 to the length of the list-1
4. create a variable smallest with initial value 'i'.
5. Create an inner Loop with a loop variable 'j' that counts from 'j+1' up to the length of the 'list-1'.
6. Inside the inner loop, if the elements at index 'j' is smaller than the element, it index smallest, then set smallest equal to 'j'.
7. After the inner loop finishes swap the elements at indexes 'i' and smallest
8. Stop.

Output:

Enter the list : 1 3 2 0 8 9 6 5 4

Sorted list : [0, 1, 2, 3, 4, 5, 6, 8, 9]

process finished.

Program No. [ ][ ] Date : [ ][ ]-[ ][ ]-[ ][ ]

Aim: Write a python program to implement selection

Program:-

```
def selection_sort(alist):
    for i in range(0,len(alist)-1):
        smallest = i
        for j in range(i+1, len(alist)):
            if alist[j] < alist[smallest]:
                smallest = j
                alist[j], alist[smallest] = alist[smallest], alist[i]
alist = input("Enter the list:")
alist = alist.split()
alist = [int(x) for x in alist]
selection_sort(alist)
print('sorted list:', end = '')
print(alist)
```

Result:

Thus, the program to implement selection sort is executed successfully and output is noted.

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Aim: To develop a python program to perform matrix addition and matrix multiplication:

Algorithm:- [Matrix Addition]

1. Start
2. Input two matrices (3x3)
3. Start loop for rows (i) and columns (j)
4. Iterate the both rows and columns and perform operation
5. Print the result

Matrix Multiplication:-

6. Input two matrices (3x3)
7. Start loop for rows 'A' and columns 'B' and rows of 'B' have 3 loops.
8. Iteration by rows of A and columns of B and by rows of B and perform operation
9. Print the result
10. Stop

Output for Addition :-

[10, 10, 10],
[10, 10, 10],
[10, 10, 10].
process finished

Output for Multiplication :-

7AB

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Program for Matrix Addition :-

```
x = [ [1,2,3],
      [4,5,6],
      [7,8,9]]

y = [[ 9,8,7],
     [ 6,5,4],
     [3,2,1]]

result = [[0,0,0],
          [0,0,0],
          [0,0,0]]

for i in range (len(x)):
   for j in range(len(x[0])):
      result[i][j] = x[i][j] + y[i][j]

for i in result
print (r)
```

program for Matrix Multiplication :-

```
A = [[12, 7, 3],
     [4,5,6]
     [7,8,9]]

B = [[ 5,8,1,2],
     [6,7,3,0]
     [4,5,9,1]].
```

Output for Multiplication:

[114, 160, 60, 27]
[74, 97, 73, 14]
[119, 157, 112, 23]

process finished

Program No. [ ][ ] Date : [ ][ ]-[ ][ ]-[ ][ ]

```
result = [[0,0,0,0],
          [0,0,0,0],
          [0,0,0,0].
for i in range (len(A)):
    for j in range (len(B[0])):
        for k in range (len(B)):
            result[i][j]+=A[i][k]* B[k][j]
for r in result:
print (r).
```

Result:
Thus, the python program to perform matrix Addition and matrix multiplication is successfully executed.

Sri Venkateswara College of Engineering & Technology

Autonomous

Department of Computer Science

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Aim: Write a python program that takes command line arguments (word count)

Algorithm:-

1) Start

2) Importing a function sys
   n = len(sys.argv

3) Read and store 'n' total arguments passed

3.1 - Read and store sys-arg[0] value as name of python script.

3.2 - Read & store end=" " value in arguments passed.

4) Read the values of i using for loop(2,n)

4.1 - Read sys.argv[i], end=" "
   Sum = 2

5) Read the values of i using for loop(1,n)
   Sum += int(sys.argv[i])

6) Print and read result as sum

7) Stop.

Program :-

```
import sys
  n = len( sys.argv)
 Print ("total arguments passed, "n)
print ("\n name of Python script", sys.argv[0])
print ("\n arguments passed", end=" ")
  for i in range(1,n):
    Print (sys.argv[i], end=" ")
     Sum = 2
```

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

```
for i in range (1,n):
    sum += int(sys.argv[i])
print("\n\n result", sum)
```

Result:

Thus, the python program to take word count is executed and outputs are noted carefully.

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Aim: To develop a python program using string slices and methods.

Algorithm:-

1. Start
2. Input the string
3. Perform the slicing on string
4. Peform the slicing method on string
5. Stop.

program:-

```
String = "Nagendra"
print (string [0:5])
print (string [-4:])
   S1 = slice (2)
   S2 = slice (-4)
   S3 = slice (1,8,2)
   S4 = slice (-1,-6,-2)
print (string [S1])
print [string [S2])
print ( string [S3])
print (string [S4])
print (String [::-1])
Plint (string. capitalize ())
print (string. count ("a"))
print (string. upper ())
```

Program No. [ ][ ] Date : [ ][ ]-[ ][ ]-[ ][ ]

```
print (string.swapcase())
print (string.split ())
print (string.replace("Nagendra","Jagadeesh"))
print (string.isalnum())
print (string.isalpha())
print (string.isdigit())
```

Result:
Thus, the python program is executed and outputs are verified Successfully.

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Aim : Implement a python program using lists and perform various Operations.

~~Algorithm~~:-

Program:-

```
# list indexing.
mylist =[' p', 'r', 'o', 'b', 'e']
print (mylist[0])
print (mylist[02])
print (mylist[-1])
print (mylist[-5])
# append ()
mylist. append (4)
print (mylist)
# extend ()
mylist .extend ([5,6,7])
print (mylist)
# insert ()
mylist. insert (3,9)
print (mylist)
# remove ()
mylist . remove ('b')
print (mylist)
# pop ()
mylist. pop(0)
print (mylist)
```

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

```
#slice
print (mylist [:4])
print (mylist [2:])
print (mylist [2:4])
print (mylist [:])
#reverse
print (mylist [::-1])
mylist.reverse()
print (mylist)
#len()
print (len (mylist))
#min() & max()
print (min ([1,2,3]))
print (max ([1,2,3]))
#count()
list = [1,2,3,3,4,4,3,1,1,3]
print (list.count(3))
#concadinate
print (mylist + list)
#multiply
print (mylist *2)
#index()
print (mylist.index('e'))
print (mylist.index('e',0,2)
print (list.sort())
```

Result: Thus, the python program using lists and performs various operations is successfully executed.

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Aim: To develop a python program to implement the dictionaries.

program:-

```
dict1 = {'Name':"mahes", "Age":20, "college": "SVCET"}
print ("dict1 :", dict1)
seq = dict1.keys()
print ("dict1.keys() :", seq)
print ("dict1.values :", dict1.values())
print ("dict1.items() :", dict1.items())
dict2 = {}
dict2 = dict2.fromkeys (seq)
print ("dict2 -from keys (seq):", dic2)
print ("dict1.get ('Name') :', dict1.get('Name',"Not present"))
dict2.clear()
print ("dict 2.clear() :", dict2)
dict2 = {"department": "CSE"}
dict2.update (dict1)
Print ("dict2. update(dict1):", dict2)
dict2. set-default ('Sex', Name): ", dict2)
dict1['Age'] = 21
print ("dict1 ['Age'] =21 ", dict())
del dict()
print(dict1)
```

Result:

Thus, the program is executed sucessfully and output is taken.

Sri Venkateswara College of Engineering & Technology

Autonomous

Department of Computer Science

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Aim: A Python Program to read a file line by line store it into an array.

Algorithm:-

1) Start
2) open the file ("test1.txt",'r')
3) content equals to file.read()
4) line equals to content.splitlines()
5) printline
6) Stop.

Program:-

```
file = open ("test1.txt",'r')
content = file.read()
line = content.splitlines()
print(line)
```

Result:

Thus, the python program to read a file line by line store into an array is successfully executed and result is taken.

Sri Venkateswara College of Engineering & Technology

Autonomous

Department of Computer Science

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Aim: The python program to find the most frequent words in a text read from a file.

Algorithm:-

1) Start
2) Open the 'test1.txt' line
2.1- frequent word equal " "
2.2 - frequency equals Zero
2.3 - words equals empty
3) Read the line from file.
   line-word = line.lower().replace(';',' ')
   replace(':',' ').split(' ')
3.1 - Read the w from line.word
   words.append(w)
4) Read i value from (0,len(words))
   count=1
4.1 - Read i value from (i+1, len(words))
   If(words[i] == words[j])
   Count = Count+1
   j=1
5) If count > frequency):
   frequency = count
   frequency_word = words[j]
6) print frequent_word
7) print str(frequency)
8) close file and stop

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

program:

```
file = open("file1.txt","r")
frequent_word = ""
frequency = 0
words = []
for line in file:
    line_word = line.lower().replace(';',"").replace(':',"").split(" ")

        for w in line_word:
            words.append(w)
        for i in range(0,len(words)):
            count = 1
        for j in range(i+1,len(words)):
            if(words[i]==words[j]):
                count = count+1
        j=1
        if(count>frequency):
            frequency = count
            frequency_word = words[j]
    print("Most repeated word :"+frequent_word)
    print("frequency:" + str(frequency))
file.close().
```

Result: Thus the python program is executed successfully.

Program No. ☐☐ Date : ☐☐-☐☐-☐☐

Aim:- A python program that demonstrates inheritances.

Algorithm:-

1) Start
2) Create parent class
3) Use function called def
3.1- print
4) declare child class
5) Use function called def
5.1 - print
6) Access the parent class function by using the child class object
7) Stop.

program:-

```
Class parent():
  def first(self):
    print("Nagendra")
  Class child(parent):
    def second(self):
      print("Jagadeesh")
      ob.child()
      ob.first()
      ob.second()
```

Result: Thus, the python program that demonstrates inheritance is successfully executed and result is noted.