AMC ENGINEERING COLLEGE

18th KM, Bannerghatta Road, Kalkere, Bangalore – 560 083
Phone : 27828655, Telefax: 27828656
E-mail :principal@amcec.edu.in, Website: www.amcec.edu.in

SEMINAR REPORT ON
PYTHON

Name : .....

USN : .....

Branch : .....

Semester : .....

LISTS

- The list is a most versatile datatype available in Python which can be written as a list of comma-separated values (items) between square brackets. Important thing about a list is that items in a list need not be of the same type.
- Creating a list is as simple as putting different comma-separated values between square brackets. For example −
- list1 = ['physics', 'chemistry', 1997, 2000]; list2 = [1, 2, 3, 4, 5]; list3 = ["a", "b", "c", "d"] Similar to string indices, list indices start at 0

Accessing Values in Lists

- To access values in lists, use the square brackets for slicing along with the index or indices to obtain value available at that index. For example −
- list1 = [‘abc', ‘def', 121, 675];
- list2 = [1, 2, 3, 4, 5, 6, 7];
- print ("list1[0]: ", list1[0])
- print ("list2[1:5]: ", list2[1:5])
- When the above code is executed, it produces the following result −
- list1[0]: abc
- list2[1:5]: [2, 3, 4, 5]

UPDATING LISTS

- list = ['physics', 'chemistry', 1997, 2000];
- print ("Value available at index 2 : ")
- print (list[2])
- list[2] = 2001;
- print("New value available at index 2 : “)
- print(list[2])
- When the above code is executed, it produces the following result −
- Value available at index 2 : 1997
- New value available at index 2 : 2001

Delete List Elements

- To remove a list element, you can use the del statement if you know exactly which element(s) you are deleting
- list1 = ['physics', 'chemistry', 1997, 2000]; print (list1)
- del list1[2];
- Print("After deleting value at index 2 : “)
- print (list1)

List Length

- To determine how many items a list have, use the len() method:
- thislist = ["apple", "banana", "cherry"]
  print(len(thislist))

Output:3

Add Items

- To add an item to the end of the list, use the append() method:
- Example
- Using the append() method to append an item:
- thislist = ["apple", "banana", "cherry"]
  thislist.append("orange")
  print(thislist)
- Output:
- ['apple', 'banana', 'cherry', 'orange']

Python Tuples

- A tuple is a collection which is ordered and **unchangeable**. In Python tuples are written with round brackets.
- Example
- Create a Tuple:
- thistuple = ("apple", "banana", "cherry")
  print(thistuple)
- Output:
- ('apple', 'banana', 'cherry')

Access Tuple Items

- You can access tuple items by referring to the index number, inside square brackets:
- Example
- Return the item in position 1:
- thistuple = ("apple", "banana", "cherry")
  print(thistuple[1])
- Output:
- banana

Loop Through a Tuple

- You can loop through the tuple items by using a for loop.
- Example
- Iterate through the items and print the values:
- thistuple = ("apple", "banana", "cherry")
  for x in thistuple:
    print(x)
- output:
- apple
  banana
  cherry

Tuple Length

- To determine how many items a list have, use the len() method:
- Example
- Print the number of items in the tuple:
- thistuple = ("apple", "banana", "cherry")
  print(len(thistuple))
- Output:
- 3

Remove Items

- **Note:** You cannot remove items in a tuple.
- Tuples are **unchangeable**, so you cannot remove items from it, but you can delete the tuple completely:
- Example
- The del keyword can delete the tuple completely:
- thistuple = ("apple", "banana", "cherry")
- del thistuple
- print(thistuple) #this will raise an error because the tuple no longer exists
- Error: name 'thistuple' is not defined

The tuple() Constructor

- It is also possible to use the tuple() constructor to make a tuple.
- Example
- Using the tuple() method to make a tuple:
- Thistuple=tuple(("apple", "banana", "cherry")) # note the double round-brackets

- print(thistuple)
- Output:
- ('apple', 'banana', 'cherry')

Python Functions

- A function is a block of code which only runs when it is called.
- You can pass data, known as parameters, into a function.
- A function can return data as a result.

Creating a Function

- In Python a function is defined using the def keyword:
- Example
- def my_function():
  print("Hello from a function")

Calling a Function

- To call a function, use the function name followed by parenthesis:
- def my_function():
  print("Hello from a function")

  my_function()
- **Output:**
- Hello from a function

Python GUI – tkinter

- Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter outputs the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.
  To create a tkinter:
- Importing the module – tkinter
- Create the main window (container)
- Add any number of widgets to the main window
- Apply the event Trigger on the widgets.

- import tkinter
- m = tkinter.Tk()
- '''
- widgets are added here
- '''
- m.mainloop()

Button

- **Button**:To add a button in your application, this widget is used.
- The general syntax is:
- w=Button(master, option=value)
- master is the parameter used to represent the parent window.
  There are number of options which are used to change the format of the Buttons. Number of options can be passed as parameters separated by commas. Some of them are listed below.
- **activebackground**: to set the background color when button is under the cursor.
- **activeforeground**: to set the foreground color when button is under the cursor.
- **bg**: to set he normal background color.
- **command**: to call a function.
- **font**: to set the font on the button label.
- **image**: to set the image on the button.
- **width**: to set the width of the button.
- **height**: to set the height of the button.

- import tkinter as tk
- r = tk.Tk()
- r.title('Counting Seconds')
- button = tk.Button(r, text='Stop', width=25, command=r.destroy)
- button.pack()
- r.mainloop()

Label

- **Label**: It refers to the display box where you can put any text or image which can be updated any time as per the code.
  The general syntax is:
- w=Label(master, option=value)
- master is the parameter used to represent the parent window.

- Example:
- from tkinter import *
- root = Tk()
- w = Label(root, text=‘python seminar')
- w.pack()
- root.mainloop()

ListBox

- **Listbox**: It offers a list to the user from which the user can accept any number of options.
  The general syntax is:
- w = Listbox(master, option=value)
- Example:
- from tkinter import *
-
- top = Tk()
- Lb = Listbox(top)
- Lb.insert(1, 'Python')
- Lb.insert(2, 'Java')
- Lb.insert(3, 'C++')
- Lb.insert(4, 'Any other')
- Lb.pack()
- top.mainloop()