Vocational Training(3rd year CSE)

Under: **Professor (Dr) Dilip Kumar**

Name: Manisha Das

Roll No: 2017UGCS010

Topic: Python programming and basics of Natural Language Processing

**DAY 1**

Installation of python and implementation of basic python programming to understand the syntax to do various operation such as print, assigning values, defining data types etc

The code is given as follows:

|  |
| --- |
| print('Hello everyone')  Hello everyone |
| '42'+5  **---------------------------------------------------------------------------**  **TypeError** Traceback (most recent call last)  **<ipython-input-2-85a5a9616fc1>** in <module>  **----> 1 '42'+5**  **TypeError**: can only concatenate str (not "int") to str |
| 'My'+' first program'  'My first program' |
| spam='hello'  spam  'hello' |
| print('Hello everyone')  myName=input()  Hello everyone  Manisha |
| print('It is good to meet you, '+ myName)  It is good to meet you, Manisha |
| len(myName)  7 |
| #swap two numbers  x=5  y=6  x,y=y,x |
| X, Y  6,5 |

**DAY 2**

Importing libraries such as numpy, pandas and implementation of them in programming to create arrays etc:

|  |
| --- |
| import numpy as np |
| a = np.arange(15).reshape(3, 5)  a  Out[1] : array([[ 0, 1, 2, 3, 4],  [ 5, 6, 7, 8, 9],  [10, 11, 12, 13, 14]]) |
| a.shape  Out[2] : (3, 5) |
| a.ndim  Out[3] : 2 |
| a.size  Out[4] : 15 |
| B = np.arange(3)  B  Out[8] : array([0, 1, 2]) |
| a = np.ones((2,3), dtype=int)  a  Out[10] : array([[1, 1, 1],  [1, 1, 1]]) |
| C = np.array([2., -1., 4.])  np.add(B, C)  #add tabhi hi hoga jab shape same hoga dono ka  Out[11] : array([2., 0., 6.]) |
| np.transpose(a)  Out[12] : array([[8., 9.],  [6., 2.]]) |

**DAY 3**

Implementation of user defined functions in python and trying them with error handling. Also trying various operations on strings and lists like reverse, length etc

* Understanding functions and locality of variables

|  |
| --- |
| #function add  def add(a,b):  c=a+b  return c |
| #function call  add('4','8')  Out[1] : '48' |
| c #it shows that c is locally defined  **------------------------------------------------------------------**  **NameError** Traceback (most recent call last)  **<ipython-input-5-c4875175c9c9>** in <module>  **----> 1** c **#it shows that c is locally defined**  **NameError**: name 'c' is not defined |
| def add(a,b):  global c  c=a+b  return c |
| add(4,8) #here c is globally defined  c  Out[3] : 12 |
| #swap function  def swap(a,b):  return(b,a) |
| #function call  swap(5,3)  Out[3] : (3, 5) |
| def local1():  x='local1'  print(x)  def local2():  x='local2'  print(x)  local1()  print(x) |
| x='global'  local2()  print(x)  local2  local1  local2  global |
| a,b=5,0  a/b  **---------------------------------------------------------------------------**  **ZeroDivisionError** Traceback (most recent call last)  **<ipython-input-14-855432946c79>** in <module>  1 a**,**b**=5,0**  **----> 2** a**/**b  **ZeroDivisionError**: division by zero |
| #Exception Handling  def divide(a,b):  try:  a/b  except ZeroDivisionError:  print('The remainder cannot be zero') |
| divide(5,0)  The remainder cannot be zero |
| #function with string  def intro(country):  print('I belong from the country '+country)  #function call  intro('India')  I belong from the country India |
| #List has no definite size, variety of data types can be stored  #List can store multiple values in ordered sequence  l=[] |
| l=[2,'d',4,'name',1.414,True] # list is ',' delimited  print(l[0])  2 |
| l[6]  **---------------------------------------------------------------------------**  **IndexError** Traceback (most recent call last)  **<ipython-input-30-cb842cca7e42>** in <module>  **----> 1** l**[6]**  **IndexError**: list index out of range |
| l.reverse() #inplace function(automatically gets assigned) |
| student\_list=[['Manisha',21,'CSE'],['Roshni',20,'ECE']] |
| #slicing of list  student\_list[:3]  Out[5] : [['Manisha', 21, 'CSE'], ['Roshni', 20, 'ECE']] |
| #list concatenation(not in place)  l+student\_list  Out[6] : [False,  1.414,  'name',  4,  'd',  2,  ['Manisha', 21, 'CSE'],  ['Roshni', 20, 'ECE']] |
| names=[]  i=0  while True:  print('enter the name!')  name=input()  if(name==''):  break  names.append(name)  i=i+1  enter the name!  Manisha  enter the name!  Roshni  enter the name!  Subhadeep  enter the name! |
| names.sort() #inplace |
| #String and tuples are immutable  s='There is no vaccine yet available for covid-19' |
| s  Out[8] : 'There is no vaccine yet available for covid-19' |
| s[0]='t'  **---------------------------------------------------------------------------**  **TypeError** Traceback (most recent call last)  **<ipython-input-70-e50119e04be2>** in <module>  **----> 1** s**[0]='t'**  **TypeError**: 'str' object does not support item assignment |
| #Tuple  t=('1','name','a',None,44) |
| t[0]='5'  **---------------------------------------------------------------------------**  **TypeError** Traceback (most recent call last)  **<ipython-input-79-e78bf5671696>** in <module>  **----> 1** t**[0]='5'**  **TypeError**: 'tuple' object does not support item assignment |
| t.append(10)  **---------------------------------------------------------------------------**  **AttributeError** Traceback (most recent call last)  **<ipython-input-80-eefa595c211a>** in <module>  **----> 1** t**.**append**(10)**  **AttributeError**: 'tuple' object has no attribute 'append' |

**DAY 4**

According to assigned task of taking two numbers as word numeric string and converting it to numbers and then adding it

* A normal addition functions for two numbers

|  |
| --- |
| def Add(a,b):  c=a+b  return c |
| Add(3,5)  output:  8 |
| print('Enter first number to add')  x=input()  print('Enter second number to add')  y=input()  no\_1=int(x)  no\_2=int(y)  Add(no\_1,no\_2)  Enter first number to add  44  Enter second number to add  10  Out[4]: 54 |

Addition of two numbers which can be either in words format or number format.

Implemented a words\_to\_number function which takes a string as an input and then it segregates both words according to several conditions as word formatting, number format, separating words as ‘and’, ‘,’, etc.

* Initializing a number system dictionary for referencing while converting words to numbers.

|  |
| --- |
| numbersystem = {  'zero': 0, 'one': 1,  'two': 2, 'three': 3, 'four': 4,  'five': 5, 'six': 6,  'seven': 7, 'eight': 8,  'nine': 9, 'ten': 10,  'eleven': 11, 'twelve': 12,  'thirteen': 13, 'fourteen': 14,  'fifteen': 15, 'sixteen': 16,  'seventeen': 17, 'eighteen': 18,  'nineteen': 19, 'twenty': 20,  'thirty': 30, 'forty': 40,  'fifty': 50, 'sixty': 60,  'seventy': 70, 'eighty': 80,  'ninety': 90, 'hundred': 100, 'thousand': 1000,  'million': 1000000, 'billion': 1000000000  } |

* Word\_to\_number\_conversion function

Below is a supplementary function number\_formation

|  |
| --- |
| def number\_formation(number\_words):  numbers = []  for number\_word in number\_words:  numbers.append(numbersystem[number\_word])  if len(numbers) == 4:  return (numbers[0] \* numbers[1]) + numbers[2] + numbers[3]  elif len(numbers) == 3:  return numbers[0] \* numbers[1] + numbers[2]  elif len(numbers) == 2:  if 100 in numbers:  return numbers[0] \* numbers[1]  else:  return numbers[0] + numbers[1]  else:  return numbers[0] |
| #function to convert a word to number  def word\_to\_number\_conversion(number\_sentence):  if type(number\_sentence) is not str:  print('Enter a valid string numeric or integer value')    number\_sentence = number\_sentence.replace('-', ' ')  #number\_sentence = number\_sentence.lower() # converting input to lowercase    #if(number\_sentence.isdigit()): # return the number if user enters number string  # return int(number\_sentence)  try:  float(number\_sentence)  return float(string)  except:  pass    #The split() method splits a string into a list.  #You can specify the separator, default separator is any whitespace.  #The strip() method removes any leading (spaces at the beginning) and trailing (spaces at the end)  #characters (space is the default leading character to remove)  split\_words = number\_sentence.strip().split() # strip extra spaces and split sentence into words    clean\_numbers = []  # removing and, & etc.  for word in split\_words:  if word in numbersystem:  clean\_numbers.append(word)    #decimal numbers to be implemented later on    billion\_index = clean\_numbers.index('billion') if 'billion' in clean\_numbers else -1  million\_index = clean\_numbers.index('million') if 'million' in clean\_numbers else -1  thousand\_index = clean\_numbers.index('thousand') if 'thousand' in clean\_numbers else -1    total\_sum=0    if len(clean\_numbers) > 0:  if len(clean\_numbers) == 1:  total\_sum += numbersystem[clean\_numbers[0]]  else:  if billion\_index > -1:  billion\_multiplier = number\_formation(clean\_numbers[0:billion\_index])  total\_sum += billion\_multiplier \* 1000000000    if million\_index > -1:  if billion\_index > -1:  million\_multiplier = number\_formation(clean\_numbers[billion\_index+1:million\_index])  else:  million\_multiplier = number\_formation(clean\_numbers[0:million\_index])  total\_sum += million\_multiplier \* 1000000    if thousand\_index > -1:  if million\_index > -1:  thousand\_multiplier = number\_formation(clean\_numbers[million\_index+1:thousand\_index])  elif billion\_index > -1 and million\_index == -1:  thousand\_multiplier = number\_formation(clean\_numbers[billion\_index+1:thousand\_index])  else:  thousand\_multiplier = number\_formation(clean\_numbers[0:thousand\_index])  total\_sum += thousand\_multiplier \* 1000  if thousand\_index > -1 and thousand\_index != len(clean\_numbers)-1:  hundreds = number\_formation(clean\_numbers[thousand\_index+1:])  elif million\_index > -1 and million\_index != len(clean\_numbers)-1:  hundreds = number\_formation(clean\_numbers[million\_index+1:])  elif billion\_index > -1 and billion\_index != len(clean\_numbers)-1:  hundreds = number\_formation(clean\_numbers[billion\_index+1:])  elif thousand\_index == -1 and million\_index == -1 and billion\_index == -1:  hundreds = number\_formation(clean\_numbers)  else:  hundreds = 0  total\_sum += hundreds      return total\_sum |

* Addition function

|  |
| --- |
| def Addition(a,b):  no\_1=word\_to\_number\_conversion(a)  no\_2=word\_to\_number\_conversion(b)  c=no\_1+no\_2  return c |
| print('Enter first number to add')  x=input()  print('Enter second number to add')  y=input()  Addition(x,y)  Enter first number to add  one hundred and one  Enter second number to add  one  Out[12]:102 |

**DAY 5**

Improving the existing program by including decimal numbers, error handling, taking input in one line, including more operations of multiplication, subtraction and division along with addition.

Also the number of numbers to perform the operations can be more than one now

* Additional function to check for float numbers and strings containing float values.

|  |
| --- |
| is\_float = False  def check\_for\_float(string):  try:  float(string)  is\_float = True  return float(string)  except:  print("Not a float")  is\_float = False |
| string = "25.010"  check\_for\_float(string)  Out[10] : 25.01 |
| def word\_to\_number\_conversion(number\_sentence):  #function to convert a word to number as above |
| word\_to\_number\_conversion("2325.141")  Out[11] : 2325.141 |
| def Addition\_of\_numbers(x):  add\_total=0  for each\_number in x:  each\_number=each\_number.strip()  print(each\_number)  add\_total+=word\_to\_number\_conversion(each\_number)  return add\_total |
| def simplifying\_input(s):  P=['hundred','thousand','million','billion','trillion']  p = s.split() # Splitting the words in s and obtaining a list containing individual words  # Now to remove the unnecessary and between or input except the word "and" which separates both words    words = [] #Initializing an empty list  w = "" #to concat all the words belonging together    #Now we recombine all the words that belong together and append them to "words" list  #so as to obtain a list containing all the individual words separately in "words" list    for index, word in enumerate(p):  # Now if the word is that "and" which is separating two words  # then we've concatenated the individual word in w  if word=="and" and p[index-1] not in P:  #ignore this #print("1. i: " + str(index) + " p2[index-1]:" + str(p2[index-1]) + " list words:" + str(words))  words.append(w)# word has been completed so now appending it  w = "" #to obtain the next word again    # Now when the word is that "and" which is between the individual word then we'll ignore it  # "YAHA WOH JO TUMNE P WAALA LOGIC DIYA THA WOH LIKH DENA MUTKU"  elif word=="and" and p[index-1] in P:  #ignore this #print("2. i: " + str(index) + " p2[index-1]:" + str(p2[index-1]) + " list words:" + str(words))  pass    elif word!="and":  #if index!=0: # ignore this 2 lines  # print("3. i: " + str(index) + " p2[index-1]:" + str(p2[index-1]) + " list words:" + str(words))  w = w + " " + word  # Now we'll append the last word obtained to the list  words.append(w)  return words |
| def splitting(x):  numeric\_values=[]  x=x.strip()  x=x.lower()  if x[0:3]=='add':  x=x[3:]  words= simplifying\_input(x)  for each\_word in words:  string=each\_word  if '+' in string:  string=string.split('+')  for word\_after\_split in string:  numeric\_values.append(word\_after\_split)  else:  numeric\_values.append(each\_word)    return numeric\_values |
| print('Enter numbers to add:')  Addition\_of\_numbers(splitting(input()))  Enter numbers to add:  add six billion forty eight million twenty thousand one hundred and forty three  six billion forty eight million twenty thousand one hundred forty three  Out[12]: 6048020143 |
| Addition\_of\_numbers(splitting(input()))  add one hundred twenty three and 200.5  one hundred twenty three  200.5  Out[14]: 323.5 |

* Now for multiplying, subtraction and division

|  |
| --- |
| def Multiplication\_of\_numbers(x):  total=1.0  for each\_number in x:  each\_number=each\_number.strip()  print(each\_number)  total\*=word\_to\_number\_conversion(each\_number)  return total |
| def Subraction\_of\_numbers(x):  total=0.0  for each\_number in x:  each\_number=each\_number.strip()  print(each\_number)  total-=word\_to\_number\_conversion(each\_number)  return total |
| def Division\_of\_numbers(x):  a,b=x  return int(a)/int(b) |
| def splitting(x,p):  numeric\_values=[]  words= simplifying\_input(x)  for each\_word in words:  string=each\_word  if p>0:  if p==1:  if '+' in string:  string=string.split('+')  for word\_after\_split in string:  numeric\_values.append(word\_after\_split)  elif ',' in string:  string=string.split(',')  for word\_after\_split in string:  numeric\_values.append(word\_after\_split)  else:  numeric\_values.append(each\_word)  if p==2:  if '\*' in string:  string=string.split('\*')  for word\_after\_split in string:  numeric\_values.append(word\_after\_split)  elif ',' in string:  string=string.split(',')  for word\_after\_split in string:  numeric\_values.append(word\_after\_split)  else:  numeric\_values.append(each\_word)  if p==3:  if '-' in string:  string=string.split('-')  for word\_after\_split in string:  numeric\_values.append(word\_after\_split)  elif ',' in string:  string=string.split(',')  for word\_after\_split in string:  numeric\_values.append(word\_after\_split)  else:  numeric\_values.append(each\_word)  if p==4:  if 'by' in string:  string=string.split('by')  for word\_after\_split in string:  numeric\_values.append(word\_after\_split)  elif 'by' in string:  string=string.split('by')  for word\_after\_split in string:  numeric\_values.append(word\_after\_split)  elif ',' in string:  string=string.split(',')  for word\_after\_split in string:  numeric\_values.append(word\_after\_split)  else:  numeric\_values.append(each\_word)    return numeric\_values    print('Enter numbers:')  x=input()  p=0  x=x.strip()  x=x.lower()  if x[0:3]=='add':  x=x[3:]  p=1  print(Addition\_of\_numbers(splitting(x,p)))  elif x[0:8]=='multiply':  x=x[8:]  p=2  print(Multiplication\_of\_numbers(splitting(x,p)))  elif x[0:8]=='subtract':  x=x[8:]  p=3  print(Subraction\_of\_numbers(splitting(x,p)))  elif x[0:6]=='divide':  x=x[6:]  p=4  print(Division\_of\_numbers(splitting(x,p)))  else:  raise ValueError('Enter the operation!') |
| Enter numbers:  multiply 2 and 3  2  3  6.0 |
| Enter numbers:  divide 6 and 3  2.0 |