|  |  |  |
| --- | --- | --- |
| **Ex.No.6** | **Text Processing using**  **String in python** | **Reg.No: URK23CS1261** |
| **13.3.24** |
| **6 A) Write a Python program that accepts a comma separated sequence of words as input and prints the unique words in sorted form (alphanumerically).**  **Sample Input: red, black, pink, green**  **Sample Output: black, green, pink, red**  **Aim:** The objective of this program is to print unique words in sorted form.  **Algorithm:**  Step 1: Start the program.  Step 2: Prompt the user to enter a list of colors separated by commas and store it in the variable 'colors'.  Step 3: Split the 'colors' string using the split(',') method and store the result in the list 'order1'.  Step 4: Use a list comprehension to strip leading and trailing spaces from each item in 'order1'.  Step 5: Convert 'order1' into a set to remove duplicate colors.  Step 6: Convert the set 'order1' back into a list to maintain the order of colors.  Step 7: Sort the list 'order1' alphabetically.  Step 8: Print "Output: " without a newline.  Step 9: Print each item in 'order1' separated by a comma and a space.  Step 10: End the program.  **Program:**  colors = input("Enter the colors: ").split(",")  order1 = [x.strip() for x in colors]  order1 = set(order1)  order1 = list(order1)  order1.sort()  print("Output: ", end ="")  print(\*order1, sep=", ")  print("╔═════════════╗\n║   Tanvik    ║\n║ URK23CS1261 ║\n╚═════════════╝")  **Output:**          **Result:** Thus, The program has successfully produced the desired output. | | |
| **6 B) Write a python program to built-in string validation methods for basic data. It can check if a string is composed of alphabetical characters, alphanumeric characters, digits, etc.**  **Aim:** The objective of this program is to string validation with alphabetical characters, alphanumeric characters, digits  **Algorithm:**  Step 1: Start the program.  Step 2: Define a string variable 's' containing the input string 'qA2'.  Step 3: Initialize boolean variables 'isal', 'isalno', 'isdi', 'islow', and 'isup' to False.  Step 4: Iterate through each character 'x' in the string 's' using a for loop.  Step 4.1: Check if the character 'x' is alphanumeric using the 'isalnum()' method.  - If True, set 'isalno' to True.  Step 4.2: Check if the character 'x' is alphabetic using the 'isalpha()' method.  - If True, set 'isal' to True.  Step 4.3: Check if the character 'x' is a digit using the 'isdigit()' method.  - If True, set 'isdi' to True.  Step 4.4: Check if the character 'x' is a lowercase letter using the 'islower()' method.  - If True, set 'islow' to True.  Step 4.5: Check if the character 'x' is an uppercase letter using the 'isupper()' method.  - If True, set 'isup' to True.  Step 5: Print the boolean values of 'isalno', 'isal', 'isdi', 'islow', and 'isup' separated by newlines.  Step 6: End the program.  **Program:**  if \_\_name\_\_ == '\_\_main\_\_':  s = 'qA2'  isal = False  isalno = False  isdi = False  islow = False  isup = False  for x in s:  if x.isalnum() == True:  isalno = True  if x.isalpha() == True:  isal = True  if x.isdigit() == True:  isdi = True  if x.islower() == True:  islow = True  if x.isupper() == True:  isup = True  print(isalno,isal, isdi, islow, isup, sep='\n')  print("╔═════════════╗\n║ Tanvik ║\n║ URK23CS1261 ║\n╚═════════════╝")  **Output:**    **Result:** Thus, The program has successfully produced the desired output.  **6 C)** **Write a Python program to find characters count of a string which are passed as list.**  **Input: St= “hello welcome” lst = [“l”, ‟w”, ‟m”, ‟e”]**  **Output:**  **l 3**  **w 1**  **m 1**  **e 3.**  **Aim:** The objective of this program is to find characters count of a string.  **Algorithm:**  Step 1: Start the program.  Step 2: Prompt the user to enter a string and store it in the variable 'string'.  Step 3: Prompt the user to enter a list of letters to find in the string, separated by spaces, and store it in the variable 'list1'.  Step 4: Strip leading and trailing spaces from 'list1' and split it into a list using the split(" ") method. Store the result in 'list1'.  Step 5: Create an empty list 'list2' to store the counts of each letter from 'list1' found in the string.  Step 6: Iterate over each letter 'x' in 'list1'.  Step 6.1: Count the occurrences of the current letter 'x' in the string using the count() method and store the count in 'list2'.  Step 7: Iterate over each letter 'letter' and its corresponding count 'no' using the zip() function on 'list1' and 'list2'.  Step 7.1: Print the letter 'letter' and its count 'no' in the format "{letter} {no}".  Step 8: End the program.  **Program:**  string = input("Enter the String: ")  list1 = (input("Enter the letters you want to find: ").strip()).split(" ")  list2 = [string.count(x) for x in list1]  for letter, no in zip(list1,list2):  print(f"{letter} {no}")  print("╔═════════════╗\n║ Tanvik ║\n║ URK23CS1261 ║\n╚═════════════╝")  **Output:**    **Result:** Thus, The program has successfully produced the desired output.  **6 D)** **Write a Python program to find the first appearance of the substring 'not' and 'bad' from a given string, if 'not' follows the 'bad', replace the whole 'not'...'bad' substring with 'good'. Return the resulting string.**  **Sample Input :**  **The song is not that bad!**  **The song is poor!**  **Sample Output:**  **The song is good!**  **The song is poor!**  **Aim:** The objective of this program is to find the first appearance of the substring and modify it.  **Algorithm:**  Step 1: Start the program.  Step 2: Prompt the user to enter two strings separated by spaces and store them in 'string1' and 'string2' respectively.  Step 3: Strip leading and trailing spaces from each input string and split them into lists of words using the split(" ") method. Store the results in 'string1' and 'string2'.  Step 4: Iterate over each string (string1 and string2) using a loop.  Step 4.1: Iterate over each word 'x' in the current string using a loop ranging from 0 to the length of the string.  Step 4.1.1: Check if the current word 'x' is equal to 'not'.  - If true, check if the word two positions ahead (x+2) is 'bad!'.  - If true, replace 'not' with 'good!', remove 'bad!' and the word after 'not' by setting them to empty strings.  Step 4.2: Print the modified string after processing all the words.  Step 5: End the program.  **Program:**  string1 = (input("Enter the String: ").strip()).split(" ")  string2 = (input("Enter the String: ").strip()).split(" ")  for string in [string1, string2]:  for x in range(0, len(string)):  if string[x] == 'not':  if string[x+2] == 'bad!':  string[x] = 'good!'  string[x+2] = ''  string[x+1] = ""  print(\*string)  print("╔═════════════╗\n║ Tanvik ║\n║ URK23CS1261 ║\n╚═════════════╝")  **Output:**    **Result:** Thus, The program has successfully produced the desired output. | | | |
| **Ex.No.7** | **Graphical User Interface** | **Reg.No: URK23CS1261** |
| **20.3.24** |
| **7 A)** **Create a GUI application to design a simple calculator or a convertor as given below.**    **Aim:** The objective of this program is to design a simple calculator or a convertor. Algorithm: Step 1: Start the program.Step 2: Create a tkinter window with dimensions 400x400.Step 3: Create functions for addition, subtraction, multiplication, and division operations.Step 4: Inside each function, retrieve values from entry widgets, perform the respective operation, and display the result in another entry widget.Step 5: Create labels and entry widgets for inputting values and displaying results.Step 6: Create buttons for each operation (+, -, \*, /) and bind them to their respective functions.Step 7: Place all widgets (labels, entry widgets, and buttons) using grid layout manager. Step 8: Run the main loop to start the tkinter application. Program: from tkinter import \*  main = Tk()  main.geometry("400x400")  def add():  print('add')  var1 = int(labl2\_var.get())  var2 = int(labl3\_var.get())  result\_var = var1+var2  entr3.delete(0, END)  entr3.insert(0, result\_var)  def sub():  var1 = int(labl2\_var.get())  var2 = int(labl3\_var.get())  result\_var = var1-var2  entr3.delete(0, END)  entr3.insert(0, result\_var)  def mul():  var1 = int(labl2\_var.get())  var2 = int(labl3\_var.get())  result\_var = var1\*var2  entr3.delete(0, END)  entr3.insert(0, result\_var)    def div():  var1 = int(labl2\_var.get())  var2 = int(labl3\_var.get())  try:  result\_var = var1/var2  except ZeroDivisionError:  print("Error /0")  else:  entr3.delete(0, END)  entr3.insert(0, result\_var)  labl1 = Label(master=main, text="Calculator", font=("Arial",26))  labl1.grid(row=0, column=0,padx=2,pady=10)  Frame1 = Frame(master=main)  Frame1.grid(row=1, column=0, sticky='nsew', padx=10, pady=10)  labl2 = Label(master=Frame1, text="Type Value 1:", font=("Arial",13))  labl2.grid(row=1, column=0,padx=20,pady=5)  labl3 = Label(master=Frame1, text="Type Value 2:", font=("Arial",13))  labl3.grid(row=2, column=0,padx=20,pady=5)  labl2\_var = StringVar()  entr1 = Entry(master=Frame1, textvariable=labl2\_var)  entr1.grid(row=1, column=1,padx=10,pady=5)  labl3\_var = StringVar()  entr2 = Entry(master=Frame1, textvariable=labl3\_var)  entr2.grid(row=2, column=1,padx=10,pady=5)  Frame1.rowconfigure(7, weight=1)  Frame2 = Frame(master=Frame1)  Frame2.grid(row=3, column=1, sticky='nsew', padx=10, pady=10)  but1 = Button(master=Frame2, text='+',bg='green', fg='white', width=3, height=1, command=add)  but2 = Button(master=Frame2, text='-',bg='green', fg='white', width=3, height=1, command=sub)  but3 = Button(master=Frame2, text='x',bg='green', fg='white', width=3, height=1, command=mul)  but4 = Button(master=Frame2, text='/',bg='green', fg='white', width=3, height=1, command=div)  but1.grid(row=0, column=0,padx=(20,5),pady=5)  but2.grid(row=0, column=1,padx=5,pady=5)  but3.grid(row=0, column=2,padx=5,pady=5)  but4.grid(row=0, column=3,padx=5,pady=5)  res = Label(master=Frame1, text="Result:", font=("Arial",13))  res.grid(row=4, column=0,padx=20,pady=5)  entr3 = Entry(master=Frame1)  entr3.grid(row=4, column=1,padx=10,pady=5) main.mainloop()print("╔═════════════╗\n║   Tanvik    ║\n║ URK23CS1261 ║\n╚═════════════╝")Output:     **Result:** Thus, The program has successfully produced the desired output. | | |