**Week 7- Programs on Functions, Recursion, Callback and Closure**

**Name: Naren Chandrashekhar**

**SRN: PES2UG20CS216**

**Section: G**

|  |  |
| --- | --- |
| Program 1 | Using Functions generate prime numbers using method of Sieve of eratosthenes. |
|  | **Program:**  **def sieve\_eratosthenes(n):**  **prime\_nums = []**  **for i in range(2,n+1):**  **if i not in prime\_nums:**  **print(i)**  **for j in range(i\*i,n+1,i):**  **prime\_nums.append(j)**  **num=int(input("Enter a number to generate prime numbers till "))**  **sieve\_eratosthenes(num)** |
|  | Output: |
| Program 2 | 1. Reverse the given string using recursion. 2. Solve tower of Hanoi problem. 3. Use recursion to raise a number to a given power n. |
|  | Program:  #a)  def reverse(string):  if len(string) == 0:  return string  else:  return reverse(string[1:]) + string[0]  Str = input("Enter a string ")  print(reverse(Str))  #b)  def towerofHanoi(n, source, destination, aux):  if n==1:  print("move disc 1 from", source, "tower", destination, "tower")  return  towerofHanoi (n-1, source, aux, destination)  print("Move disc ",n,"from",source,"tower",destination,"tower")  towerofHanoi(n-1,aux,destination,source)  num = int(input("Enter number of discs "))  towerofHanoi(num,"left","right","middle")  #c)  def power(n,b):  if n==0:  return 1  else:  return b\*power(n-1,b)  n = int(input("Enter the value of power "))  b = int(input("Enter the value of base "))  a = power(n,b)  print(a) |
|  | Output: |
| Program 3 | 1. Use callback to find sum ,double and triple the given number. |
|  | **Program:**  **def double(x):**  **x = 2\*x**  **print("The value of the given number, doubled is",x)**  **def triple(y):**  **y = 3\*y**  **print("The value of the given number, tripled is",y)**  **def find(z,num):**  **z(num)**  **a = int(input("Enter a given number "))**  **find(double,a)**  **find(triple,a)** |
|  |  |
| Program 4 | 1. Increment a given number by 5 for n number of times using Closure. 2. Find Nth root of a given Number By Closure |
|  | **Program:**  **def increment(x,n):**  **def increment1():**  **print(x+(n\*5))**  **return increment1**  **x = int(input("Enter the given number "))**  **n = int(input("Enter the number of times the number should be incremented by 5 "))**  **y = increment(x,n)**  **y()**  **def root(u,v):**  **def nth\_root():**  **print((u)\*\*(1/v))**  **return nth\_root**  **u = int(input("Enter the given number "))**  **v = int(input("Enter the root value "))**  **w = root(u,v)**  **w()** |
|  | **Output:** |
| Program 5 | Using Concept of decorator find nth fibonnaci number. |
|  | **Program**  **def func(outer):**  **x = outer()**  **print("nth Fibonnaci number= ",x)**  **@func**  **def outer():**  **a=1**  **b=1**  **n = int(input("Enter n to find nth Fibonnaci number: "))**  **for i in range(2,n):**  **a,b = b,b+a**  **return b**  **print(outer)** |
|  | **Output:** |
| Program 6 | Design a GUI based Calculator using Tkinter. |
|  | **Program:**  **from tkinter import \***  **#creating buttons**  **root = Tk() #creates a window**  **root.title("Simple Calculator")**  **e = Entry(root , width = 35, borderwidth = 5)**  **e.grid(row = 0 , column = 0 , columnspan = 3 , padx = 10, pady = 10)**  **def click(number):**  **current = e.get()**  **e.delete(0,END)**  **e.insert(0, str(current) + str(number))**  **def clear():**  **e.delete(0,END)**  **def add():**  **f\_n = e.get()**  **global f\_num**  **global math**  **math = "addition"**  **f\_num = int(f\_n)**  **e.delete(0,END)**  **def sub():**  **f\_n = e.get()**  **global f\_num**  **global math**  **math = "subtraction"**  **f\_num = int(f\_n)**  **e.delete(0,END)**  **def mul():**  **f\_n = e.get()**  **global f\_num**  **global math**  **math = "multiplication"**  **f\_num = int(f\_n)**  **e.delete(0,END)**  **def div():**  **f\_n = e.get()**  **global f\_num**  **global math**  **math = "division"**  **f\_num = int(f\_n)**  **e.delete(0,END)**    **def equal():**  **s\_num = int(e.get())**  **e.delete(0, END)**  **if math == "addition":**  **e.insert(0 , f\_num + s\_num)**  **elif math == "subtraction":**  **e.insert(0 , f\_num - s\_num)**  **elif math == "multiplication":**  **e.insert(0 , f\_num \* s\_num)**  **elif math == "division":**  **e.insert(0 , f\_num / s\_num)**  **#define buttons**  **bt1 = Button(root, text = "1", padx = 40 , pady = 20, command = lambda : click(1))#callback**  **bt2 = Button(root, text = "2", padx = 40 , pady = 20, command = lambda : click(2))#callback**  **bt3 = Button(root, text = "3", padx = 40 , pady = 20, command = lambda : click(3))#callback**  **bt4 = Button(root, text = "4", padx = 40 , pady = 20, command = lambda : click(4))#callback**  **bt5 = Button(root, text = "5", padx = 40 , pady = 20, command = lambda : click(5))#callback**  **bt6 = Button(root, text = "6", padx = 40 , pady = 20, command = lambda : click(6))#callback**  **bt7 = Button(root, text = "7", padx = 40 , pady = 20, command = lambda : click(7))#callback**  **bt8 = Button(root, text = "8", padx = 40 , pady = 20, command = lambda : click(8))#callback**  **bt9 = Button(root, text = "9", padx = 40 , pady = 20, command = lambda : click(9))#callback**  **bt0 = Button(root, text = "0", padx = 40 , pady = 20, command = lambda : click(0))#callback**  **btadd = Button(root, text = "+", padx = 39 , pady = 20, command = add)#callback**  **btsub = Button(root, text = "-", padx = 41 , pady = 20, command = sub)#callback**  **btmul = Button(root, text = "\*", padx = 40 , pady = 20, command = mul)#callback**  **btdiv = Button(root, text = "/", padx = 41 , pady = 20, command = div)#callback**  **btEquals = Button(root, text = "=", padx = 90 , pady = 20, command = equal)#callback**  **btClear = Button(root, text = "Clear", padx = 79 , pady = 20, command = clear)#callback**  **bt1.grid(row = 3, column = 0)**  **bt2.grid(row = 3, column = 1)**  **bt3.grid(row = 3, column = 2)**  **bt4.grid(row = 2, column = 0)**  **bt5.grid(row = 2, column = 1)**  **bt6.grid(row = 2, column = 2)**  **bt7.grid(row = 1, column = 0)**  **bt8.grid(row = 1, column = 1)**  **bt9.grid(row = 1, column = 2)**  **bt0.grid(row = 4, column = 0 )**  **btadd.grid(row = 5, column = 0 )**  **btEquals.grid(row = 5, column = 1, columnspan = 2)**  **btClear.grid(row = 4, column = 1, columnspan = 2)**  **btsub.grid(row = 6, column = 0 )**  **btmul.grid(row = 6, column = 1 )**  **btdiv.grid(row = 6, column = 2 )**  **root.mainloop()** |
|  | Output: |
|  |  |