|  |  |
| --- | --- |
| 1 a | Generate the following pattern for a given value of n. (say n = 4)  1 0 0 0  0 1 0 0  0 0 1 0  0 0 0 1  **1:PROGRAM TO DISPLAY IDENTITY MATRIX OF GIVEN SIZE:**  n=int(input("enter the number"))  i=0  while i in range(0,n):  j=0  while j in range(0,n):  if i==j:  print("1",end="\t")  else:  print("0",end="\t")  j+=1  i+=1  print("\n") |
|  |  |
| 1 b | Create a directory PAP with 10 files. Each file should contain different info. Write a program to find biggest file in the directory PAP.  import glob  import os  l=list(filter(os.path.isfile, glob.glob("\*")))  print(l)  n=[]  for i in range(0,5):  big=l[0]  for j in l:  if os.path.getsize(big)<os.path.getsize(j):  big=j  n.append(big)  l.remove(big)  print()  print("Sorted list of the Files")  for i in n:  print(i+"\t:\t"+str(os.path.getsize(i))+"b") |
|  |  |
| 1 c | Define a procedure, greatest, that takes as input a list of positive numbers, and returns the greatest number in that list. If the input list is empty, the output should be 0. |
|  | def greatest(p):  if(p==[]):  return 0  else:  i=0  greatest = p[i]  for a in p:  if(a>greatest):  greatest = a  return greatest  def main():  list = []  while True:  n = int(input('Enter either 0 to stop input process or a positive number to add to list: '))  if n == 0:  break  list.append(n)  print('The greatest number in the list', list, 'is', greatest(list))  main() |
|  |  |
| 2 a | Create a directory PAP with 10 files. Remove all empty files in this directory. |
|  |  |
| 2 b | Write a program to read the name and date of birth from the keyboard and insert into the table. Display the contents and verify the same at the backend also.  import pymysql  con=pymysql.connect()  cur=con.cursor()  name=input("Enter name : ")  dob=input("Enter dob(yyyy,mm,dd) : ")  cur.execute('use test')  cur.execute('''insert into details values (%s,%s)''',(name,dob))  cur.execute('select \* from details')  rows=cur.fetchall()  for row in rows:  print(row)  con.commit()  cur.close()  con.close() |
|  |  |
| 2 C | Write a program which takes 2 digits, X,Y as input and generates a 2-dimensional array. The element value in the i-th row and j-th column of the array should be i\*j.  Note: i=0,1.., X-1; j=0,1,¡­Y-1.  Example  Suppose the following inputs are given to the program:  3,5  Then, the output of the program should be:  [[0, 0, 0, 0, 0], [0, 1, 2, 3, 4], [0, 2, 4, 6, 8]]  Hints:  Note: In case of input data being supplied to the question, it should be assumed to be a console input in a comma-separated form. |
|  | input\_str = input()  dimensions=[int(x) for x in input\_str.split(',')]  rowNum=dimensions[0]  colNum=dimensions[1]  multilist = [[0 for col in range(colNum)] for row in range(rowNum)]  for row in range(rowNum):  for col in range(colNum):  multilist[row][col]= row\*col  print (multilist) |
| 3 a | Walk through this given string and find all tags.  <test>  <sub>PAP</sub>  <date>  <dd>30</dd>  <mm>9</mm>  <yy>2013</yy>  </date>  </test>  **9:PROGRAM TO SEPARATE TAGS USING RE IN THE GIVEN PROGRAM:**  // Wrong solution does not display all tags  import re  p1='[A-Z 0-9]'  p='<.+?>'  s='<text><sub>PAP</sub><abc></abc><date><dd>30</dd><mm>9</mm><yy>2013</yy></date></text>'  l=s.split(' ')  for i in l:  l1=re.split(p1,i)  for j in l1:  a=re.search(p,j)  if a:  print(a.group()) |
| 3 b | Create a list of elements. Use a dict to find the frequency of each element in the list.  **15:PROGRAM TO COUNT THE OCCURRENCE OF NUMBERS IN A LIST:**  from collections import Counter  a=[]  n=input("Enter the range of the list")  i=1  while (i<=n):  num=input("Number : ")  a.append(num)  i+=1  print "The list :"  print a  print "Dictionary :"  c = Counter(a)  print c |
| 3 c | Design a code snippet in program simply takes a list of files on the command line, and lists the contents of each, with lines numbered. |
|  | #!/usr/bin/python  # Print the contents of the files listed on the command line.  import sys  for fn in sys.argv[1:]:  try:  fin = open(fn, 'r')  except:  (type, detail) = sys.exc\_info()[:2]  print "\n\*\*\* %s: %s: %s \*\*\*" % (fn, type, detail)  continue  print "\n\*\*\* Contents of", fn, "\*\*\*"    # Print the file, with line numbers.  lno = 1  while 1:  line = fin.readline()  if not line: break;  print '%3d: %-s' % (lno, line[:-1])  lno = lno + 1  fin.close()  print |
| 4 a | Validate a given date. Check also for leap year.  w="22-4-2008"  print(w)  a='11-22-2012'  print(a)  b=a.split('-')  print(b)  c=int(b[0])  d=int(b[1])  e=int(b[2])  if(c<=31 and e<=2013):  print("valid date")  if(e%4==0):  print("leap year")  else:  print("not a leap year")  else:  print("invalid date") |
| 4 b | Write a program to find all numbers which are odd and which are palindromes between a pair of numbers. Use list comprehension. |
| 4 c | Write a program that accepts a sentence and calculate the number of letters and digits. |
|  | s = input()  d={"DIGITS":0, "LETTERS":0}  for c in s:  if c.isdigit():  d["DIGITS"]+=1  elif c.isalpha():  d["LETTERS"]+=1  else:  pass  print ("LETTERS", d["LETTERS"])  print ("DIGITS", d["DIGITS"]) |
| 5 a | Write a function to check whether a given number is prime or not. Use this function to find all prime numbers in a given range  **8:PROGRAM TO FIND PRIME NUMBERS BETWEEN GIVEN RANGE:**  import math  def prime(a):  for i in range(2,(int)(math.sqrt(a)+1)):  if ((int)(a%i==0)):  return 0  return 1  a=int(input("Enter the starting number"))  b=int(input("Enter the ending number"))  m=0  print("The list is:")  for i in range(a,(b+1)):  m=prime(i)  if m==1:  print(i) |
| 5 b | create a module MyMath.py which contains functions to compute  i) area of a sphere given the radius  ii) volume of a sphere given the radius  iii) radius of a sphere given the volume  Test these functions in the module. Write a client program to use these functions by importing.  **6:PROGRAM TO FIND AREA,VOLUME AND RADIUS:**  import MyMath  t=1  while(t==1):  choice=int(input("Enter 1 for computing area \nEnter 2 for computing volume \nEnter 3 for computing radius \nEnter any other value to terminate\n"))  if(choice==1):  r1=float(input("Enter radius to calculate area:\n"))  MyMath.area(r1)  elif(choice==2):  r2=float(input("Enter radius to calculate volume:\n"))  MyMath.volume(r2)  elif(choice==3):  v1=float(input("Enter volume to calculate radius:\n"))  MyMath.radius(v1)  else:  t=0  print("Thanks for trying\n")  module MyMath.py:  import math  def area(r):  a=4\*math.pi\*r\*r  print ("Area", a)  def volume(r):  v=4/3\*math.pi\*r\*r\*r;  print ("Volume",v)  def radius(v):  r=math.pow((v\*3/(4\*math.pi)),1/3)  print ("Radius",r) |
| 5 C | Write a program that accepts a sentence and calculate the number of upper case letters and lower case letters. |
|  | s = input()  d={"UPPER CASE":0, "LOWER CASE":0}  for c in s:  if c.isupper():  d["UPPER CASE"]+=1  elif c.islower():  d["LOWER CASE"]+=1  else:  pass  print ("UPPER CASE", d["UPPER CASE"])  print ("LOWER CASE", d["LOWER CASE"]) |
| 6 a | Write a function which takes two arguments say a and b. Decorate to check whether a is less than b.  **2:PROGRAM TO FIND THE BIGGEST OF TWO:**  a=int(input("Enter a "))  b=int(input("Enter b "))  def dec(func):  def f(\*args):  print(".....................")  func(\*args)  print(".....................")  return f  @dec  def compare(a,b):  a=int(a)  b=int(b)  if a<b:  print(a,"is less than",b)  elif a>b:  print(a,"is greater than",b)  else:  print(a,"equals",b)  compare(a,b) |
| 6 b | Write a program whether a given character occurs again in the same string anywhere in the string  **16: PROGRAM TO CHECK WHETHER GIVEN CHARACTER OCCURS AGAIN IN THE STRING.**  import re  sub=input("Enter the string")  pat=input("Enter character")  m=re.findall(pat,sub)  if len(m)>=2:  print("Character occurs more then once")  elif len(m)==1:  print("Character occurs only once")  else:  print("Character not found") |
| 6 C | Write a program that computes the net amount of a bank account based a transaction log from console input. The transaction log format is shown as following:  D 100  W 200  ….  D means deposit while W means withdrawal. |
|  | import sys  netAmount = 0  while True:  s = input()  if not s:  break  values = s.split(" ")  operation = values[0]  amount = int(values[1])  if operation=="D":  netAmount+=amount  elif operation=="W":  netAmount-=amount  else:  pass  print (netAmount) |
| 7 a | Write a program to find all persons whose birth day falls in a given month.  hint: use month(dob) to get the month  import pymysql  con=pymysql.connect()  cur=con.cursor()  val=input("Enter a month to check : ")  cur.execute('use test')  cur.execute('select name from details where month(dob)= %s'%val)  print('name')  rows=cur.fetchall()  for row in rows:  for val in row:  print(val,end='\t')  print()  con.commit()  cur.close()  con.close() |
| 7 b | Find the 5 biggest files in the directory /bin. Store the result in a list. Sort the list  import os  import glob  os.chdir("/New folder")  filedict=dict()  for files in glob.glob("\*"):  filedict[files]=os.path.getsize(files)  sf=(sorted(filedict.items(),key=lambda y:y[1],reverse=True))  for i in range(5):  print(sf[i]) |
| 7 C | A website requires the users to input username and password to register. Write a program to check the validity of password input by users.  Following are the criteria for checking the password:  1. At least 1 letter between [a-z]  2. At least 1 number between [0-9]  3. At least 1 letter between [A-Z]  4. At least 1 character from [$#@]  5. Minimum length of transaction password: 6  6. Maximum length of transaction password: 12  Your program should accept a sequence of comma separated passwords and will check them according to the above criteria. Passwords that match the criteria are to be printed, each separated by a comma. |
|  | import re  value = []  items=[x for x in input().split(',')]  for p in items:  if len(p)<6 or len(p)>12:  continue  else:  pass  if not re.search("[a-z]",p):  continue  elif not re.search("[0-9]",p):  continue  elif not re.search("[A-Z]",p):  continue  elif not re.search("[$#@]",p):  continue  elif re.search("\s",p):  continue  else:  pass  value.append(p)  print (",".join(value)) |
| 8 a | Write a program which generates an HTML form which reads two numbers and calls another program (as part of action) which displays the numbers and their sum |
| 8 b | Write a program to find the # of occurrences of keywords IF DO THEN ENDDO ENDIF in a given sentence / program.  **17:PROGRAM TO FIND # OF OCCURENCES OF KEYWORDS IF DO THEN ENDDO ENDIF.**  import re  regex=(r'IF|DO|ENDDO|THEN|ENDIF')  rc=re.compile(regex)  dict={}  dict['IF']=0  dict['DO']=0  dict['THEN']=0  dict['ENDDO']=0  dict['ENDIF']=0  string=''' IF x<8  IF  ENDDO  x square  ENDDO  THEN x+8  ENDIF'''  for i in rc.findall(string):  if i=='IF':  a=dict[i]  a+=1  dict[i]=a  elif i=='DO':  a=dict[i]  a+=1  dict[i]=a  elif i=='THEN':  a=dict[i]  a+=1  dict[i]=a  elif i=='ENDDO':  a=dict[i]  a+=1  dict[i]=a  elif i=='ENDIF':  a=dict[i]  a+=1  dict[i]=a  print(dict) |
| 8 C | Write a program to compute the frequency of the words from the input. The output should output after sorting the key alphanumerically. |
|  | // sort not working so please refer to ur soln  freq = {} # frequency of words in text  line = raw\_input()  for word in line.split():  freq[word] = freq.get(word,0)+1  words = freq.keys()  words.sort()  for w in words:  print ("%s:%d" % (w,freq[w])) |
| 9 a | Create a class called MyStack which supports push and pop operations.  **5:PROGRAM TO DO STACK OPERATIONS:**  class mystack:    def \_\_init\_\_(self):  self.st = []  def push(self,p):  self.st.insert(0,p)    def pop(self):  try:  self.st.pop(0)  except:  print("Stack is empty")  def display(self):  print("Array elements")  print(self.st)  c1=mystack()  while(True):  ch=int(input("1.Push to stack\n2.Delete from stack\n3.Display elements\n4.exit\n"))  if (ch==1):  a=input("Enter element")  c1.push(a)  if(ch==2):  c1.pop()  if(ch==3):  c1.display()  if(ch==4):  print("Exiting")  exit() |
| 9 b | Generate all perfect squares which are palindromes and so are their square roots less than or equal to given number n. |
| 9 c | 1. Write a program which uses map() and filter() to make a list whose elements are square of even number in [1,2,3,4,5,6,7,8,9,10]. 2. Write a program which can filter() to make a list whose elements are even number between 1 and 20 (both included). |
|  | (i) li = [1,2,3,4,5,6,7,8,9,10]  evenNumbers = map(lambda x: x\*\*2, filter(lambda x: x%2==0, li))  print (evenNumbers)  (ii) evenNumbers = filter(lambda x: x%2==0, range(1,21))  print evenNumbers |
| 10 a | Write a program to find union and intersection of given lists L1 and L2.  **12:PROGRAM TO FIND UNION ,INTERSECTION OF GIVEN LIST:**  n1=input("enter list 1 range\n")  i=1  j=1  a=[]  b=[]  while(i<=n1):  num=input("numbr:")  a.append(num)  i+=1  print "the list 1 :"  print a  n2=input("enter list 2 range\n")  while(j<=n2):  num=input("numbr:")  b.append(num)  j+=1  print "the list 2 :"  print b  def intersect(a, b):  return list(set(a) & set(b))  print("the two lists are\n")  print(a)  print("\n")  print(b)  print("\n the intersected elements are")  print intersect(a, b)  def union(a,b):  return list(set(a) | set(b))  print("\n union is ")  print union(a,b) |
| 10 b | Write a program to sum the elements of an array (list) using 4 threads. Let each thread add quarter of the array. Assume that the size of the array is a multiple of 4.  **21:PROGRAM TO SUM THE ELEMENTS OF AN ARRAY USING THREADS.**  import threading  a=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16]  j=0  k=4  sum1=0  class mythread(threading.Thread):  def \_\_init\_\_(self,low,high):  threading.Thread.\_\_init\_\_(self)  self.low=low  self.high=high  self.total=0  def run(self):  for i in range(self.low,self.high):  self.total+=a[i]  for i in range(4):  thread=mythread(j,k)  thread.start()  thread.join()  sum1+=thread.total  print("Thread",i,":",thread.total)  j=k  k=k+4  print("sum:",sum1) |
|  |  |
| 11 a | Write a graphic based program to add two numbers and display the result. |
| 11 b | Generate the following pattern for a given value of n. (say n = 4)  1 = 1  1 + 2 = 3  1 + 2 + 3 = 6  1 + 2 + 3 + 4 = 10  **10:PROGRAM TO FIND THE SUM OF THE FOLLOWING FORM:**  m=input("enter number:\n")  h=range(1,m+1)  b='='  for i in range(m):  a=''  for j in range(i+1):  s=str(h[j])  a=a+s  sum=(str.join("+",a))  print(sum+b+str(eval(sum))) |
|  |  |
| 12 a | Generate Pascal triangle. if n = 4, output should be  1  1 1  1 2 1  1 3 3 1  1 4 6 4 1  **14:PROGRAM TO GENERATE THE FOLLOWING PASCAL TRIANGLE:**  n=input("enter the value of n")  for i in range(n):  value=1  plist=[value]  for j in range(i):  value=value\*(i-j)\*1/(j+1)  plist.append(int(value))  print("{:^40s}".format(plist))  print ("done") |
| 12 b | write a function which gets # of strings using variable # of arguments and returns a list of strings which are palindromes  **19:PROGRAM WHICH GETS NUMBER OF STRINGS USING VARIABLE NUMBER OF ARGUMENTS & RETURNS A LIST OF STRINGS WHICH ARE PALINDROMES.**  def reversenumber(n,partial=0):  if n==0:  return partial  return reversenumber(n/10,partial\*10+n%10)  trial=int(input("Enter the number"))  if reversenumber(trial)==trial:  print("Its a palindrome!!")  else:  print("Is not a palindrome") |
| 12 c | The Fibonacci Sequence is computed based on the following formula:  f(n)=0 if n=0  f(n)=1 if n=1  f(n)=f(n-1)+f(n-2) if n>1  Please write a program using list comprehension to print the Fibonacci Sequence in comma separated form with a given n input by console. |
|  | def f(n):  if n == 0: return 0  elif n == 1: return 1  else: return f(n-1)+f(n-2)  n=int(raw\_input())  values = [str(f(x)) for x in range(0, n+1)]  print (",".join(values)) |
|  |  |
| 13 a | Write a program to sum the elements of an array (list) using 4 processes. Let each thread add quarter of the array. Assume that the size of the array is a multiple of 4. |
| 13 b | Write a program with appropriate rules to tokenize the following as follows:  a --- b => (a, ID) (--, OP) (-, OP) (b, ID)  a << b => (a, ID) (<<, OP) (b, ID)  a < b - c => (a, ID) (< , OP) (b, ID) (-, op) (c, ID)  **18:PROGRAM TO TOKENIZE THE FOLLOWING AS FOLLOWS.**  a---b->(a,ID) (--,op) (-,op) (b,ID)  a<<b->(a,ID) (<<,op) (b,ID)  a<b-c->(a, ID) (<,op) (b,ID) (-,op) (c,ID)  import re  r1=(r'[a-z]---[a-z]')  r2=(r'[a-z]<<[a-z]')  r3=(r'[a-z]<[a-z]-[a-z]')  string='''a---b\na<<<b\na<b-c'''  print(string)  rc1=re.compile(r1)  rc2=re.compile(r2)  rc3=re.compile(r3)  print("a--b")  for i in rc1.findall(string):  print(i[0],'ID')  print(i[1]+i[2],'OP')  print(i[3],'OP')  print(i[4],'ID')  print("a<<b")  for i in rc2.findall(string):  print(i[0],'ID')  print(i[1]+i[2],'OP')  print(i[3],'ID')  print("a<b-c")  for i in rc3.findall(string):  print(i[0],'ID')  print(i[1],'OP')  print(i[2],'ID')  print(i[3],'OP')  print(i[4],'ID') |
| 13 c | Write a program using generator to print the even numbers between 0 and n in comma separated form while n is input by console. |
|  | def EvenGenerator(n):  i=0  while i<=n:  if i%2==0:  yield i  i+=1  n=int(raw\_input())  values = []  for i in EvenGenerator(n):  values.append(str(i))  print (",".join(values)) |