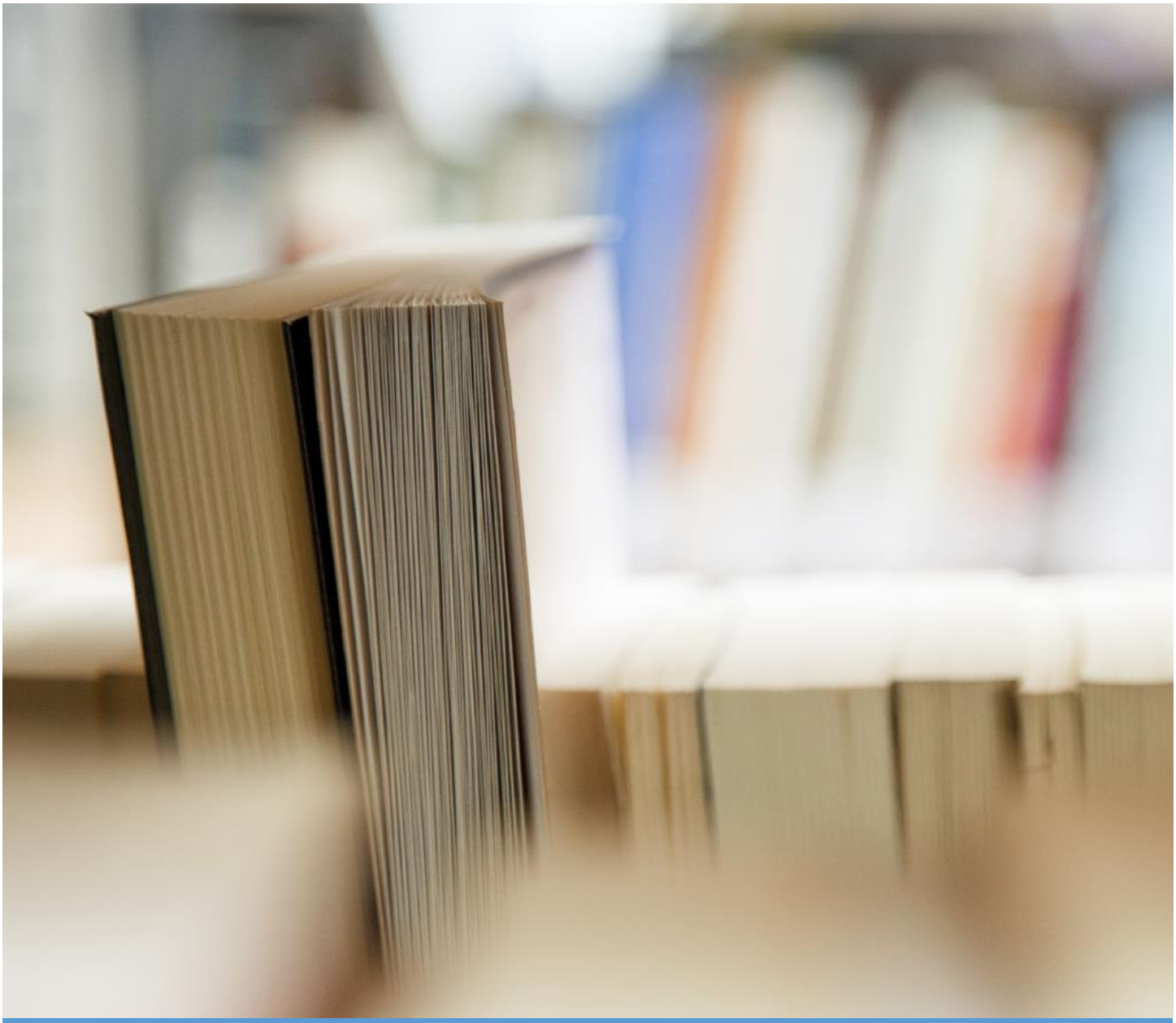

PYTHON EXERCISE

AMAR PANCHAL – 9821601163

1. Write a Python program to print the following string in a specific format (see the output).

Output :

Twinkle, twinkle, little star,
How I wonder what you are!
Up above the world so high,
Like a diamond in the sky.
Twinkle, twinkle, little star,
How I wonder what you are

Solution:-

```
print("Twinkle, twinkle, little star, \n\tHow I wonder what you are! \n\t\tUp above the world so high, \n\t\tLike a diamond in the sky. \nTwinkle, twinkle, little star, \n\tHow I wonder what you are!")
```


2. Write a Python program to get the Python version you are using.

Solution:-

```
import sys
print("Python version")
print (sys.version)
print("Version info.")
print (sys.version_info)
```

3. Write a Python program to display the current date and time.

Solution:-

```
import datetime
now = datetime.datetime.now()
print ("Current date and time : ")
print (now.strftime("%Y-%m-%d %H:%M:%S"))
```

4. Write a Python program which accepts the radius of a circle from the user and compute the area.

Output :

r = 1.1

Area = 3.8013271108436504

Solution:-

```
from math import pi
r = float(input ("Input the radius of the circle : "))
print ("The area of the circle with radius " + str(r) + " is: " + str(pi * r**2))
```

5. Write a Python program which accepts the user's first and last name and print them in reverse order with a space between them.

Solution:-

```
fname = input("Input your First Name : ")
lname = input("Input your Last Name : ")
print ("Hello  " + lname + " " + fname)
```


6. Write a Python program which accepts a sequence of comma-separated numbers from user and generate a list and a tuple with those numbers.

Output :

List : ['3', ' 5', ' 7', ' 23']

Tuple : ('3', ' 5', ' 7', ' 23')

Solution:-

```
values = input("Input some comma seprated numbers : ")
list = values.split(",")
tuple = tuple(list)
print('List : ',list)
print('Tuple : ',tuple)
```

7. Write a Python program to accept a filename from the user and print the extension of that.

Solution:-

```
filename = input("Input the Filename: ")
f_extns = filename.split(".")
print ("The extension of the file is : " + repr(f_extns[-1]))
```

8. Write a Python program to display the first and last colors from the following list.

Solution:-

```
color_list = ["Red","Green","White" ,"Black"]
print( "%s %s"%(color_list[0],color_list[-1]))
```

9. Write a Python program to display the examination schedule.

Solution:-

```
exam_st_date = (11,12,2014)
print( "The examination will start from : %i / %i / %i"%exam_st_date)
```


10. Write a Python program that accepts an integer (n) and computes the value of n+nn+nnn.

Solution:-

```
a = int(input("Input an integer : "))
n1 = int( "%s" % a )
n2 = int( "%s%s" % (a,a) )
n3 = int( "%s%s%s" % (a,a,a) )
print (n1+n2+n3)
```

11. Write a Python program to print the documents (syntax, description etc.) of Python built-in function(s).

Solution:-

```
print(abs.__doc__)
```

12. Write a Python program to print the calendar of a given month and year.
Note : Use 'calendar' module.

Solution:-

```
import calendar
```

```
y = int(input("Input the year : "))
m = int(input("Input the month : "))
print(calendar.month(y, m))
```

13. Write a Python program to print the following here document.

Solution:-

```
print("""
a string that you "don't" have to escape
This
is a  ..... multi-line
heredoc string --------> example
""")
```


14. Write a Python program to calculate number of days between two dates.
Sample dates : (2014, 7, 2), (2014, 7, 11)
Expected output : 9 days

Solution:-

```
from datetime import date
f_date = date(2014, 7, 2)
l_date = date(2014, 7, 11)
delta = l_date - f_date
print(delta.days)
```

15. Write a Python program to get the volume of a sphere with radius 6.

Solution:-

```
pi = 3.1415926535897931
r= 6.0
V= 4.0/3.0*pi* r**3
print('The volume of the sphere is: ',V)
```

16. Write a Python program to get the difference between a given number and 17, if the number is greater than 17 return double the absolute difference.

Solution:-

```
def difference(n):
  if n <= 17:
    return 17 - n
  else:
    return (n - 17) * 2

print(difference(22))
print(difference(14))
```

17. Write a Python program to test whether a number is within 100 of 1000 or 2000.

Solution:-

```
def near_thousand(n):
    return ((abs(1000 - n) <= 100) or (abs(2000 - n) <= 100))
print(near_thousand(1000))
print(near_thousand(900))
print(near_thousand(800))
print(near_thousand(2200))
```

18. Write a Python program to calculate the sum of three given numbers, if the values are equal then return three times of their sum.

Solution:-

```
def sum_thrice(x, y, z):

  sum = x + y + z

  if x == y == z:
```

```
    sum = sum * 3
    return sum

print(sum_thrice(1, 2, 3))
print(sum_thrice(3, 3, 3))
```

19. Write a Python program to get a new string from a given string where "Is" has been added to the front. If the given string already begins with "Is" then return the string unchanged.

Solution:-

```
def new_string(str):
  if len(str) >= 2 and str[:2] == "Is":
    return str
  return "Is" + str

print(new_string("Array"))
print(new_string("IsEmpty"))
```


20. Write a Python program to get a string which is n (non-negative integer) copies of a given string.

Solution:-

```
def larger_string(str, n):
  result = ""
  for i in range(n):
    result = result + str
  return result

print(larger_string('abc', 2))
print(larger_string('.py', 3))
```

21. Write a Python program to find whether a given number (accept from the user) is even or odd, print out an appropriate message to the user.

Solution:-

```
num = int(input("Enter a number: "))
mod = num % 2
if mod > 0:
   print("This is an odd number.")
else:
   print("This is an even number.")
```

22. Write a Python program to count the number 4 in a given list.

Solution:-

```
def list_count_4(nums):
  count = 0
  for num in nums:
    if num == 4:
      count = count + 1

  return count

print(list_count_4([1, 4, 6, 7, 4]))
print(list_count_4([1, 4, 6, 4, 7, 4]))
```


23. Write a Python program to get the n (non-negative integer) copies of the first 2 characters of a given string. Return the n copies of the whole string if the length is less than

Solution:-

```
def substring_copy(str, n):
  flen = 2
  if flen > len(str):
    flen = len(str)
```

```
    substr = str[:flen]

    result = ""
    for i in range(n):
      result = result + substr
    return result
print(substring_copy('abcdef', 2))
print(substring_copy('p', 3));
```

24. Write a Python program to test whether a passed letter is a vowel or not.

Solution:-

```
def is_vowel(char):
   all_vowels = 'aeiou'
   return char in all_vowels
print(is_vowel('c'))
print(is_vowel('e'))
```


25. Write a Python program to check whether a specified value is contained in a group of values.

Test Data :

3 -> [1, 5, 8, 3] : True

-1 -> [1, 5, 8, 3] : False

Solution:-

```
def is_group_member(group_data, n):
   for value in group_data:
      if n == value:
         return True
   return False
print(is_group_member([1, 5, 8, 3], 3))
print(is_group_member([5, 8, 3], -1))
```

26. Write a Python program to create a histogram from a given list of integers.
Solution:-

```
def histogram( items ):
   for n in items:
      output = ''
      times = n
      while( times > 0 ):
       output += '*'
       times = times - 1
      print(output)

histogram([2, 3, 6, 5])
```


27. Write a Python program to concatenate all elements in a list into a string and return it.
Solution:-

```
def concatenate_list_data(list):
   result= ''
   for element in list:
      result += str(element)
   return result

print(concatenate_list_data([1, 5, 12, 2]))
```

28. Write a Python program to print all even numbers from a given numbers list in the same order and stop the printing if any numbers that come after 237 in the sequence.
Sample numbers list :

```
numbers = [
  386, 462, 47, 418, 907, 344, 236, 375, 823, 566, 597, 978, 328, 615, 953, 345,
  399, 162, 758, 219, 918, 237, 412, 566, 826, 248, 866, 950, 626, 949, 687, 217,
  815, 67, 104, 58, 512, 24, 892, 894, 767, 553, 81, 379, 843, 831, 445, 742, 717,
  958,743, 527]
```

Solution:-

```
numbers = [
  386, 462, 47, 418, 907, 344, 236, 375, 823, 566, 597, 978, 328, 615, 953,
345,
  399, 162, 758, 219, 918, 237, 412, 566, 826, 248, 866, 950, 626, 949, 687,
217,
  815, 67, 104, 58, 512, 24, 892, 894, 767, 553, 81, 379, 843, 831, 445, 742,
717,
  958,743, 527
  ]

for x in numbers:
  if x == 237:
    print(x)
    break;
  elif x % 2 == 0:
    print(x)
```


29. Write a Python program to print out a set containing all the colors from color_list_1 which are not present in color_list_2.
Test Data :
color_list_1 = set(["White", "Black", "Red"])
color_list_2 = set(["Red", "Green"])
Expected Output :
{'Black', 'White'}

Solution:-

```
color_list_1 = set(["White", "Black", "Red"])
color_list_2 = set(["Red", "Green"])

print(color_list_1.difference(color_list_2))
```

30. Write a Python program that will accept the base and height of a triangle and compute the area.

Solution:-

```
b = int(input("Input the base : "))
h = int(input("Input the height : "))

area = b*h/2

print("area = ", area)
```


31. Write a Python program to compute the greatest common divisor (GCD) of two positive integers.

Solution:-

```
def gcd(x, y):
   gcd = 1

   if x % y == 0:
      return y

   for k in range(int(y / 2), 0, -1):
      if x % k == 0 and y % k == 0:
         gcd = k
         break
   return gcd
```

```
print(gcd(12, 17))
print(gcd(4, 6))
```

32. Write a Python program to get the least common multiple (LCM) of two positive integers.

Solution:-

```
def lcm(x, y):
  if x > y:
     z = x
  else:
     z = y

  while(True):
     if((z % x == 0) and (z % y == 0)):
        lcm = z
        break
     z += 1

  return lcm
print(lcm(4, 6))
print(lcm(15, 17))
```


33. Write a Python program to sum of three given integers. However, if two values are equal sum will be zero.

Solution:-

```
def sum(x, y, z):
   if x == y or y == z or x==z:
      sum = 0
   else:
      sum = x + y + z
   return sum
```

```
print(sum(2, 1, 2))
print(sum(3, 2, 2))
print(sum(2, 2, 2))
print(sum(1, 2, 3))
```

34. Write a Python program to sum of two given integers. However, if the sum is between 15 to 20 it will return 20.

Solution:-

```
def sum(x, y):
  sum = x + y
  if sum in range(15, 20):
    return 20
  else:
    return sum

print(sum(10, 6))
print(sum(10, 2))
print(sum(10, 12))
```


35. Write a Python program that will return true if the two given integer values are equal or their sum or difference is 5.

Solution:-

```
def test_number5(x, y):
  if x == y or abs(x-y) == 5 or (x+y) == 5:
    return True
  else:
    return False

print(test_number5(7, 2))
print(test_number5(3, 2))
```

```
print(test_number5(2, 2))
```

36. Write a Python program to add two objects if both objects are an integer type.

Solution:-

```
def add_numbers(a, b):
   if not (isinstance(a, int) and isinstance(b, int)):
      raise TypeError("Inputs must be integers")
   return a + b

print(add_numbers(10, 20))
```

37. Write a Python program to display your details like name, age, address in three different lines.

Solution:-

```
def personal_details():
   name, age = "Simon", 19
   address = "Bangalore, Karnataka, India"
   print("Name: {}\nAge: {}\nAddress: {}".format(name, age, address))

personal_details()
```

38. Write a Python program to solve (x + y) * (x + y).

Test Data : x = 4, y = 3

Expected Output : (4 + 3) ^ 2) = 49

Solution:-

```
x, y = 4, 3
result = x * x + 2 * x * y + y * y
```

```
print("({} + {}) ^ 2) = {}".format(x, y, result))
```

39. Write a Python program to compute the future value of a specified principal amount, rate of interest, and a number of years.

Test Data : amt = 10000, int = 3.5, years = 7
Expected Output : 12722.79

Solution:-

```
amt = 10000
int = 3.5
years = 7

future_value  = amt*((1+(0.01*int)) ** years)
print(round(future_value,2))
```


40. Write a Python program to compute the distance between the points (x1, y1) and (x2, y2).

Solution:-

```
import math
p1 = [4, 0]
p2 = [6, 6]
distance = math.sqrt( ((p1[0]-p2[0])**2)+((p1[1]-p2[1])**2) )

print(distance)
```

41. Write a Python program to check whether a file exists.

Solution:-

```
import os.path
open('abc.txt', 'w')
```

```
print(os.path.isfile('abc.txt'))
```

42. Write a Python program to determine if a Python shell is executing in 32bit or 64bit mode on OS.

Solution:-

```
# For 32 bit it will return 32 and for 64 bit it will return 64
import struct
print(struct.calcsize("P") * 8)
```

43. Write a Python program to get OS name, platform and release information.

Solution:-

```
import platform
import os
print(os.name)
print(platform.system())
print(platform.release())
```


44. Write a Python program to locate Python site-packages.

Solution:-

```
import site;
print(site.getsitepackages())
```

45. Write a python program to call an external command in Python.

Solution:-

```
from subprocess import call
call(["ls", "-l"])
```

46. Write a python program to get the path and name of the file that is currently executing.

Solution:-

```
import os
print("Current File Name : ",os.path.realpath(__file__))
```

47. Write a Python program to find out the number of CPUs using.

Solution:-

```
import multiprocessing
print(multiprocessing.cpu_count())
```

48. Write a Python program to parse a string to Float or Integer.

Solution:-

```
n = "246.2458"
print(float(n))
print(int(float(n)))
```


49. Write a Python program to list all files in a directory in Python.

Solution:-

```
from os import listdir
from os.path import isfile, join
files_list = [f for f in listdir('/home/students') if isfile(join('/home/students',
f))]
print(files_list);
```

50. Write a Python program to print without newline or space.

Solution:-

```
for i in range(0, 10):
   print('*', end="")
print("\n")
```

51. Write a Python program to determine profiling of Python programs.

Note: A profile is a set of statistics that describes how often and for how long various parts of the program executed. These statistics can be formatted into reports via the pstats module.

Solution:-

```
import cProfile
def sum():
   print(1+2)
cProfile.run('sum()')
```

52. Write a Python program to print to stderr.

Solution:-

```
from __future__ import print_function
import sys

def eprint(*args, **kwargs):
   print(*args, file=sys.stderr, **kwargs)

eprint("abc", "efg", "xyz", sep="--")
```


53. Write a python program to access environment variables.

Solution:-

```
import os
# Access all environment variables
print('*---------------------------------*')
print(os.environ)
print('*---------------------------------*')
# Access a particular environment variable
print(os.environ['HOME'])
print('*---------------------------------*')
print(os.environ['PATH'])
```

```
print('*---------------------------------*')
```

54. Write a Python program to get the current username

Solution:-

```
import getpass
print(getpass.getuser())
```

55. Write a Python to find local IP addresses using Python's stdlib

Solution:-

```
import socket
print([l for l in ([ip for ip in
socket.gethostbyname_ex(socket.gethostname())[2]
if not ip.startswith("127.")][:1], [[(s.connect(('8.8.8.8', 53)),
s.getsockname()[0], s.close()) for s in [socket.socket(socket.AF_INET,
socket.SOCK_DGRAM)]][0][1]]) if l][0][0])
```


56. Write a Python program to get height and width of the console window.

Solution:-

```
def terminal_size():
   import fcntl, termios, struct
   th, tw, hp, wp = struct.unpack('HHHH',
      fcntl.ioctl(0, termios.TIOCGWINSZ,
      struct.pack('HHHH', 0, 0, 0, 0)))
   return tw, th

print('Number of columns and Rows: ',terminal_size())
```

57. Write a program to get execution time for a Python method.

Solution:-

```
import time
def sum_of_n_numbers(n):
   start_time = time.time()
   s = 0
   for i in range(1,n+1):
      s = s + i
   end_time = time.time()
   return s,end_time-start_time

n = 5
print("\nTime to sum of 1 to ",n," and required time to calculate is
:",sum_of_n_numbers(n))
```

58. Write a python program to find the sum of the first n positive integers.

Solution:-

```
n = int(input("Input a number: "))
sum_num = (n * (n + 1)) / 2
print(sum_num)
```

59. Write a Python program to convert height (in feet and inches) to centimeters.

Solution:-

```
print("Input your height: ")
h_ft = int(input("Feet: "))
h_inch = int(input("Inches: "))

h_inch += h_ft * 12
h_cm = round(h_inch * 2.54, 1)

print("Your height is : %d cm." % h_cm)
```

60. Write a Python program to calculate the hypotenuse of a right angled triangle.

Solution:-

```
from math import sqrt
print("Input lengths of shorter triangle sides:")
a = float(input("a: "))
b = float(input("b: "))

c = sqrt(a**2 + b**2)
print("The length of the hypotenuse is", c )
```

61. Write a Python program to convert the distance (in feet) to inches, yards, and miles.

Solution:-

```
d_ft = int(input("Input distance in feet: "))
d_inches = d_ft * 12
d_yards = d_ft / 3.0
d_miles = d_ft / 5280.0

print("The distance in inches is %i inches." % d_inches)
print("The distance in yards is %.2f yards." % d_yards)
print("The distance in miles is %.2f miles." % d_miles)
```


62. Write a Python program to convert all units of time into seconds.

Solution:-

```
days = int(input("Input days: ")) * 3600 * 24
hours = int(input("Input hours: ")) * 3600
minutes = int(input("Input minutes: ")) * 60
seconds = int(input("Input seconds: "))

time = days + hours + minutes + seconds
```

```
print("The  amounts of seconds", time)
```

63. Write a Python program to get an absolute file path.
Solution:-

```
def absolute_file_path(path_fname):
    import os
    return os.path.abspath('path_fname')
print("Absolute file path: ",absolute_file_path("test.txt"))
```

64. Write a Python program to get file creation and modification date/times.
Solution:-

```
import os.path, time
print("Last modified: %s" % time.ctime(os.path.getmtime("test.txt")))
print("Created: %s" % time.ctime(os.path.getctime("test.txt")))
```


65. Write a Python program to convert seconds to day, hour, minutes and seconds.
Solution:-

```
time = float(input("Input time in seconds: "))
day = time // (24 * 3600)
time = time % (24 * 3600)
hour = time // 3600
time %= 3600
minutes = time // 60
time %= 60
seconds = time
print("d:h:m:s-> %d:%d:%d:%d" % (day, hour, minutes, seconds))
```

66. Write a Python program to calculate body mass index.
Solution:-

```
height = float(input("Input your height in Feet: "))
weight = float(input("Input your weight in Kilogram: "))
print("Your body mass index is: ", round(weight / (height * height), 2))
```

67. Write a Python program to convert pressure in kilopascals to pounds per square inch, a millimeter of mercury (mmHg) and atmosphere pressure.

Solution:-

```
kpa = float(input("Input pressure in in kilopascals> "))
psi = kpa / 6.89475729
mmhg = kpa * 760 / 101.325
atm = kpa / 101.325
print("The pressure in pounds per square inch: %.2f psi"  % (psi))
print("The pressure in millimeter of mercury: %.2f mmHg" % (mmhg))
print("Atmosphere pressure: %.2f atm." % (atm))
```


68. Write a Python program to calculate the sum of the digits in an integer.

Solution:-

```
num = int(input("Input a four digit numbers: "))
x  = num //1000
x1 = (num - x*1000)//100
x2 = (num - x*1000 - x1*100)//10
x3 = num - x*1000 - x1*100 - x2*10
print("The sum of digits in the number is", x+x1+x2+x3)
```

69. Write a Python program to sort three integers without using conditional statements and loops.

Solution:-

```
x = int(input("Input first number: "))
y = int(input("Input second number: "))
z = int(input("Input third number: "))
```

```
a1 = min(x, y, z)
a3 = max(x, y, z)
a2 = (x + y + z) - a1 - a3
print("Numbers in sorted order: ", a1, a2, a3)
```

70. Write a Python program to sort files by date.

Solution:-

```
import glob
import os

files = glob.glob("*.txt")
files.sort(key=os.path.getmtime)
print("\n".join(files))
```


71. Write a Python program to get a directory listing, sorted by creation date.

Solution:-

```
from stat import S_ISREG, ST_CTIME, ST_MODE
import os, sys, time

#Relative or absolute path to the directory
dir_path = sys.argv[1] if len(sys.argv) == 2 else r'.'

#all entries in the directory w/ stats
data = (os.path.join(dir_path, fn) for fn in os.listdir(dir_path))
data = ((os.stat(path), path) for path in data)

# regular files, insert creation date
data = ((stat[ST_CTIME], path)
        for stat, path in data if S_ISREG(stat[ST_MODE]))
```

```
for cdate, path in sorted(data):
   print(time.ctime(cdate), os.path.basename(path))
```

72. Write a Python program to get the details of math module.

Solution:-

```
# Imports the math module
import math
#Sets everything to a list of math module
math_ls = dir(math) #
print(math_ls)
```

73. Write a Python program to calculate midpoints of a line.

Solution:-

```
print('\nCalculate the midpoint of a line :')

x1 = float(input('The value of x (the first endpoint) '))
y1 = float(input('The value of y (the first endpoint) '))

x2 = float(input('The value of x (the first endpoint) '))
y2 = float(input('The value of y (the first endpoint) '))

x_m_point = (x1 + x2)/2
y_m_point = (y1 + y2)/2
print();
print("The midpoint of line is :")
print( "The midpoint's x value is: ",x_m_point)
print( "The midpoint's y value is: ",y_m_point)
print();
```


74. Write a Python program to hash a word.

Solution:-

```
soundex=[0,1,2,3,0,1,2,0,0,2,2,4,5,5,0,1,2,6,2,3,0,1,0,2,0,2]

word=input("Input the word be hashed: ")

word=word.upper()

coded=word[0]

for a in word[1:len(word)]:
   i=65-ord(a)
   coded=coded+str(soundex[i])
print()
print("The coded word is: "+coded)
print()
```


75. Write a Python program to get the copyright information.

Solution:-

```
import sys
print("\nPython Copyright Information")
print(sys.copyright)
print()
```

76. Write a Python program to get the command-line arguments (name of the script, the number of arguments, arguments) passed to a script.

Solution:-

```
import sys
print("This is the name/path of the script:"),sys.argv[0]
print("Number of arguments:",len(sys.argv))
print("Argument List:",str(sys.argv))
```

77. Write a Python program to test whether the system is a big-endian platform or little-endian platform.

Solution:-

```
import sys
print()
if sys.byteorder == "little":
   #intel, alpha
   print("Little-endian platform.")
else:
   #motorola, sparc
   print("Big-endian platform.")
print()
```


78. Write a Python program to find the available built-in modules.

Solution:-

```
import sys
import textwrap
module_name = ', '.join(sorted(sys.builtin_module_names))
print(textwrap.fill(module_name, width=70))
```

79. Write a Python program to get the size of an object in bytes.

Solution:-

```
import sys
str1 = "one"
str2 = "four"
str3 = "three"
print()
print("Memory size of '"+str1+"' = "+str(sys.getsizeof(str1))+ " bytes")
print("Memory size of '"+str2+"' = "+str(sys.getsizeof(str2))+ " bytes")
print("Memory size of '"+str3+"' = "+str(sys.getsizeof(str3))+ " bytes")
```

```
print()
```

80. Write a Python program to get the current value of the recursion limit.
Solution:-

```
import sys
print()
print("Current value of the recursion limit:")
print(sys.getrecursionlimit())
print()
```

81. Write a Python program to concatenate N strings.
Solution:-

```
list_of_colors = ['Red', 'White', 'Black']
colors = '-'.join(list_of_colors)
print()
print("All Colors: "+colors)
print()
```


82. Write a Python program to calculate the sum over a container.
Solution:-

```
s = sum([10,20,30])
print("\nSum of the container: ", s)
print()
```

83. Write a Python program to test whether all numbers of a list is greater than a certain number.
Solution:-

```
num = [2,3,4]
print()
print(all(x > 1 for x in num))
print(all(x > 4 for x in num))
print()
```

84. Write a Python program to count the number occurrence of a specific character in a string.

Solution:-

```
s = "The quick brown fox jumps over the lazy dog."
print()
print(s.count("q"))
print()
```

85. Write a Python program to check whether a file path is a file or a directory.

Solution:-

```
import os
path="abc.txt"
if os.path.isdir(path):
   print("\nIt is a directory")
elif os.path.isfile(path):
   print("\nIt is a normal file")
else:
   print("It is a special file (socket, FIFO, device file)" )
print()
```

86. Write a Python program to get the ASCII value of a character.

Solution:-

```
print()
print(ord('a'))
print(ord('A'))
```

```
print(ord('1'))
print(ord('@'))
print()
```

87. Write a Python program to get the size of a file.

Solution:-

```
import os
file_size = os.path.getsize("abc.txt")
print("\nThe size of abc.txt is :",file_size,"Bytes")
print()
```

88. Given variables x=30 and y=20, write a Python program to print "30+20=50".

Solution:-


```
x = 30
y = 20
print("\n%d+%d=%d" % (x, y, x+y))
print()
```

89. Write a Python program to perform an action if a condition is true.

Solution:-

```
n=1
if n == 1:
   print("\nFirst day of a month")
print()
```

90. Write a Python program to create a copy of its own source code.

Solution:-

```
print()
print((lambda str='print(lambda str=%r: (str %% str))()': (str % str))())
print()
```

91. Write a Python program to swap two variables.

Solution:-

```
a = 30
b = 20
print("\nBefore swap a = %d and b = %d" %(a, b))
a, b = b, a
print("\nAfter swaping a = %d and b = %d" %(a, b))
print()
```

92. Write a Python program to define a string containing special characters in various forms.

Solution:-

```
print()
print("\#{'}${\"}@/")
print("\#{'}${"'"'}@/")
print(r"""\#{'}${"}@/""")
print('\#{\'}${"}@/')
print('\#{'"'"'}${"}@/')
print(r'''\#{'}${"}@/''')
print()
```


93. Write a Python program to get the identity of an object.

Solution:-

```
obj1 = object()
obj1_address = id(obj1)
print()
print(obj1_address)
print()
```

94. Write a Python program to convert a byte string to a list of integers.

Solution:-

```
x = b'Abc'
print()
print(list(x))
print()
```

95. Write a Python program to check whether a string is numeric.

Solution:-

```
str = 'a123'
#str = '123'
try:
   i = float(str)
except (ValueError, TypeError):
   print('\nNot numeric')
print()
```


96. Write a Python program to print the current call stack.

Solution:-

```
import traceback
print()
def f1():return abc()
def abc():traceback.print_stack()
f1()
print()
```

97. Write a Python program to list the special variables used within the language.

Solution:-

```
s_var_names = sorted((set(globals().keys()) | set(__builtins__.__dict__.keys()))
- set('_ names i'.split()))
print()
```

```
print( '\n'.join(' '.join(s_var_names[i:i+8]) for i in range(0, len(s_var_names), 8)) )
print()
```

98. Write a Python program to get the system time.

Note : The system time is important for debugging, network information, random number seeds, or something as simple as program performance.

Solution:-

```
import time
print()
print(time.ctime())
print()
```


99. Write a Python program to clear the screen or terminal.

Solution:-

```
import os
import time
# for windows
# os.system('cls')
os.system("ls")
time.sleep(2)
# Ubuntu version 10.10
os.system('clear')
```

100. Write a Python program to get the name of the host on which the routine is running.

Solution:-

```
import socket
host_name = socket.gethostname()
```

```
print()
print("Host name:", host_name)
print()
```

101. Write a Python program to access and print a URL's content to the console.

Solution:-

```
from http.client import HTTPConnection
conn = HTTPConnection("example.com")
conn.request("GET", "/")
result = conn.getresponse()
# retrieves the entire contents.
contents = result.read()
print(contents)
```


102. Write a Python program to get system command output.

Solution:-

```
import subprocess
# file and directory listing
returned_text = subprocess.check_output("dir", shell=True,
universal_newlines=True)
print("dir command to list file and directory")
print(returned_text)
```

103. Write a Python program to extract the filename from a given path.

Solution:-

```
import os
print()
print(os.path.basename('/users/system1/student1/homework-1.py'))
print()
```

104. Write a Python program to get the effective group id, effective user id, real group id, a list of supplemental group ids associated with the current process.
Note: Availability: Unix.

Solution:-

```
import os
print("\nEffective group id: ",os.getegid())
print("Effective user id: ",os.geteuid())
print("Real group id: ",os.getgid())
print("List of supplemental group ids: ",os.getgroups())
print()
```

105. Write a Python program to get the users environment.

Solution:-

```
import os
print()
print(os.environ)
print()
```


106. Write a Python program to divide a path on the extension separator.

Solution:-

```
import os.path
for path in [ 'test.txt', 'filename', '/user/system/test.txt', '/', '' ]:
   print('"%s" :' % path, os.path.splitext(path))
```

107. Write a Python program to retrieve file properties.

Solution:-

```
import os.path
import time

print('File        :', __file__)
print('Access time  :', time.ctime(os.path.getatime(__file__)))
```

```
print('Modified time:', time.ctime(os.path.getmtime(__file__)))
print('Change time  :', time.ctime(os.path.getctime(__file__)))
print('Size        :', os.path.getsize(__file__))
```

108. Write a Python program to find path refers to a file or directory when you encounter a path name.

Solution:-

```
import os.path

for file in [ __file__, os.path.dirname(__file__), '/', './broken_link']:
    print('File       :', file)
    print('Absolute    :', os.path.isabs(file))
    print('Is File?    :', os.path.isfile(file))
    print('Is Dir?     :', os.path.isdir(file))
    print('Is Link?    :', os.path.islink(file))
    print('Exists?     :', os.path.exists(file))
    print('Link Exists?:', os.path.lexists(file))
```


109. Write a Python program to check if a number is positive, negative or zero.

Solution:-

```
num = float(input("Input a number: "))
if num > 0:
   print("It is positive number")
elif num == 0:
   print("It is Zero")
else:
   print("It is a negative number")
```

110. Write a Python program to get numbers divisible by fifteen from a list using an anonymous function.

Solution:-

```
num_list = [45, 55, 60, 37, 100, 105, 220]
# use anonymous function to filter
result = list(filter(lambda x: (x % 15 == 0), num_list))
print("Numbers divisible by 15 are",result)
```

111. Write a Python program to make file lists from current directory using a wildcard.

Solution:-

```
import glob
file_list = glob.glob('*.*')
print(file_list)
```

112. Write a Python program to remove the first item from a specified list.

Solution:-


```
color = ["Red", "Black", "Green", "White", "Orange"]
print("\nOriginal Color: ",color)
del color[0]
print("After removing the first color: ",color)
print()
```

113. Write a Python program to input a number, if it is not a number generate an error message.

Solution:-

```
while True:
   try:
      a = int(input("Input a number: "))
      break
   except ValueError:
      print("\nThis is not a number. Try again...")
      print()
```

114. Write a Python program to filter the positive numbers from a list.

Solution:-

```
nums = [34, 1, 0, -23]
print("Original numbers in the list: ",nums)
new_nums = list(filter(lambda x: x >0, nums))
print("Positive numbers in the list: ",new_nums)
```

115. Write a Python program to compute the product of a list of integers (without using for loop).

Solution:-

```
from functools import reduce
nums = [10, 20, 30,]
nums_product = reduce( (lambda x, y: x * y), nums)
print("Product of the numbers : ",nums_product)
```


116. Write a Python program to print Unicode characters.

Solution:-

```
str = u'\u0050\u0079\u0074\u0068\u006f\u006e
\u0045\u0078\u0065\u0072\u0063\u0069\u0073\u0065\u0073 \u002d
\u0077\u0033\u0072\u0065\u0073\u006f\u0075\u0072\u0063\u0065'
print()
print(str)
print()
```

117. Write a Python program to prove that two string variables of same value point same memory location.

Solution:-

```
str1 = "Python"
str2 = "Python"

print("\nMemory location of str1 =", hex(id(str1)))
```

```
print("Memory location of str2 =", hex(id(str2)))
print()
```

118. Write a Python program to create a bytearray from a list.

Solution:-

```
print()
nums = [10, 20, 56, 35, 17, 99]
# Create bytearray from list of integers.
values = bytearray(nums)
for x in values: print(x)
print()
```

119. Write a Python program to display a floating number in specified numbers.

Solution:-


```
order_amt = 212.374
print('\nThe total order amount comes to %f' % order_amt)
print('The total order amount comes to %.2f' % order_amt)
print()
```

120. Write a Python program to format a specified string to limit the number of characters to 6.

Solution:-

```
str_num = "1234567890"
print()
print('%.6s' % str_num)
print()
```

121. Write a Python program to determine whether variable is defined or not.

Solution:-

```
try:
  x = 1
```

```
except NameError:
  print("Variable is not defined....!")
else:
  print("Variable is defined.")
try:
  y
except NameError:
  print("Variable is not defined....!")
else:
  print("Variable is defined.")
```

122. Write a Python program to empty a variable without destroying it.
Sample data: n=20
d = {"x":200}
Expected Output : 0
{}

Solution:-

```
n = 20
d = {"x":200}
l = [1,3,5]
t= (5,7,8)
print(type(n)())
print(type(d)())
print(type(l)())
print(type(t)())
```

123. Write a Python program to determine the largest and smallest integers, longs, floats.

Solution:-

```
import sys
```

```
print("Float value information: ",sys.float_info)
print("\nInteger value information: ",sys.int_info)
print("\nMaximum size of an integer: ",sys.maxsize)
```

124. Write a Python program to check whether multiple variables have the same value.

Solution:-

```
x = 20
y = 20
z = 20
if x == y == z == 20:
   print("All variables have same value!")
```

125. Write a Python program to sum of all counts in a collections?

Solution:-

```
import collections
num = [2,2,4,6,6,8,6,10,4]
print(sum(collections.Counter(num).values()))
```

126. Write a Python program to get the actual module object for a given object.

Solution:-

```
from inspect import getmodule
from math import sqrt
print(getmodule(sqrt))
```

127. Write a Python program to check whether an integer fits in 64 bits.

Solution:-

```
int_val = 30
if int_val.bit_length() <= 63:
   print((-2 ** 63).bit_length())
   print((2 ** 63).bit_length())
```

128. Write a Python program to check whether lowercase letters exist in a string.
Solution:-

```
str1 = 'A8238i823acdeOUEI'
print(any(c.islower() for c in str1))
```

129. Write a Python program to add trailing and leading zeroes to a string.
Solution:-

```
str1='122.22'
print("Original String: ",str1)
print("\nAdded trailing zeros:")
str1 = str1.ljust(8, '0')
print(str1)
str1 = str1.ljust(10, '0')
print(str1)
print("\nAdded leading zeros:")
str1='122.22'
str1 = str1.rjust(8, '0')
print(str1)
str1 = str1.rjust(10, '0')
print(str1)
```


130. Write a Python program to use double quotes to display strings.
Solution:-

```
import json
print(json.dumps({'Alex': 1, 'Suresh': 2, 'Agnessa': 3}))
```

131. Write a Python program to split a variable length string into variables.
Solution:-

```
var_list = ['a', 'b', 'c']
```

```
x, y, z = (var_list + [None] * 3)[:3]
print(x, y, z)
var_list = [100, 20.25]
x, y = (var_list + [None] * 2)[:2]
print(x, y)
```

132. Write a Python program to list home directory without absolute path.
Solution:-

```
import os.path
print(os.path.expanduser('~'))
```

133. Write a Python program to calculate the time runs (difference between start and current time) of a program.
Solution:-


```
from timeit import default_timer
def timer(n):
    start = default_timer()
    # some code here
    for row in range(0,n):
        print(row)
    print(default_timer() - start)

timer(5)
timer(15)
```

134. Write a Python program to input two integers in a single line.
Solution:-

```
print("Input the value of x & y")
x, y = map(int, input().split())
print("The value of x & y are: ",x,y)
```

135. Write a Python program to print a variable without spaces between values.
Sample value : x =30
Expected output : Value of x is "30"
Solution:-

```
x = 30
print('Value of x is "{}"'.format(x))
```

136. Write a Python program to find files and skip directories of a given directory.
Sample data : x=12
Expected output : 00001100
0000001100
Solution:-

```
import os
print([f for f in os.listdir('/home/students') if
os.path.isfile(os.path.join('/home/students', f))])
```


137. Write a Python program to extract single key-value pair of a dictionary in variables.
Solution:-

```
d = {'Red': 'Green'}
(c1, c2), = d.items()
print(c1)
print(c2)
```

138. Write a Python program to convert true to 1 and false to 0.
Solution:-

```
x = 'true'
x = int(x == 'true')
print(x)
```

```
x = 'abcd'
x = int(x == 'true')
print(x)
```

139. Write a Python program to valid a IP address.

Solution:-

```
import socket
addr = '127.0.0.2561'
try:
   socket.inet_aton(addr)
   print("Valid IP")
except socket.error:
   print("Invalid IP")
```


140. Write a Python program to convert an integer to binary keep leading zeros.

Sample data : x=12

Expected output : 00001100

0000001100

Solution:-

```
x = 12
print(format(x, '08b'))
print(format(x, '010b'))
```

141. Write a python program to convert decimal to hexadecimal.

Sample decimal number: 30, 4

Expected output: 1e, 04

Solution:-

```
x = 30
print(format(x, '02x'))
x = 4
print(format(x, '02x'))
```

142. Write a Python program to find the operating system name, platform and platform release date.

Solution:-

```
import os, platform
print("Operating system name:")
print(os.name)
print("Platform name:")
print(platform.system())
print("Platform release:")
print(platform.release())
```

143. Write a Python program to determine if the python shell is executing in 32bit or 64bit mode on operating system.

Solution:-

```
import struct
print(struct.calcsize("P") * 8)
```

144. Write a Python program to check whether variable is of integer or string.

Solution:-

```
print(isinstance(25,int) or isinstance(25,str))
print(isinstance([25],int) or isinstance([25],str))
print(isinstance("25",int) or isinstance("25",str))
```

145. Write a Python program to test if a variable is a list or tuple or a set.

Solution:-

```
#x = ['a', 'b', 'c', 'd']
#x = {'a', 'b', 'c', 'd'}
x = ('tuple', False, 3.2, 1)
if type(x) is list:
    print('x is a list')
elif type(x) is set:
```

```
    print('x is a set')
elif type(x) is tuple:
    print('x is a tuple')
else:
    print('Neither a list or a set or a tuple.')
```

146. Write a Python program to find the location of Python module sources.

Solution:-

```
 import sys
print("\nList of directories in sys module:")
print(sys.path)
print("\nList of directories in os module:")
import os
print(os.path)
```


147. Write a Python function to check whether a number is divisible by another number. Accept two integers values form the user.

Solution:-

```
def multiple(m, n):
        return True if m % n == 0 else False

print(multiple(20, 5))
print(multiple(7, 2))
```

148. Write a Python function to find the maximum and minimum numbers from a sequence of numbers.

Note: Do not use built-in functions.

Solution:-

```
def max_min(data):
  l = data[0]
  s = data[0]
```

```
 for num in data:
   if num> l:
     l = num
   elif num< s:
       s = num
 return l, s

print(max_min([0, 10, 15, 40, -5, 42, 17, 28, 75]))
```

149. Write a Python function that takes a positive integer and returns the sum of the cube of all the positive integers smaller than the specified number.

Solution:-

```
def sum_of_cubes(n):
  n -= 1
  total = 0
  while n > 0:
    total += n * n * n
    n -= 1
  return total
print("Sum of cubes: ",sum_of_cubes(3))
```


150. Write a Python function to find a distinct pair of numbers whose product is odd from a sequence of integer values.

Solution:-

```
def odd_product(nums):
 for i in range(len(nums)):
  for j in range(len(nums)):
   if  i != j:
     product = nums[i] * nums[j]
     if product & 1:
       return True
       return False
```


```
dt1 = [2, 4, 6, 8]
dt2 = [1, 6, 4, 7, 8]
print(dt1, odd_product(dt1));
print(dt2, odd_product(dt2));
```