**LAB1**

**Đinh Trọng Đạt – 1914775**

**Github:** [lap-trinh-python/Lab/Lab01 at main · Dat0309/lap-trinh-python (github.com)](https://github.com/Dat0309/lap-trinh-python/tree/main/Lab/Lab01)

**Bài 1. Hoàn thành 25/150 bài tập cơ bản:**

Ex1.

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!")

Ex2.

import platform

print(platform.python\_version())

Ex3.

import datetime

curDate = datetime.datetime.now()

print('Current date and time is: \n', curDate.strftime("%Y-%m-%d %H:%M:%S"))

Ex4.

from math import pi

r = float(input("Nhap ban kinh cua hinh tron: "))

area = pi\*r\*\*2

print(f"Dien tich cua hinh tron co ban kinh {r} la: {area}")

Ex5.

first\_name = input("Nhap ho cua ban: ")

last\_name = input("Nhap ten cua ban: ")

print(f"Hello {last\_name} {first\_name}")

Ex6.

values = input("Input some comma seprated numbers : ")

list = values.split(",")

tuple = tuple(list)

print('List : ',list)

print('Tuple : ',tuple)

Ex7.

filename = input("Input the FileName: ")

file\_extention = filename.split(".")

print(f"The extension of the file is: {repr(file\_extention[-1])}")

Ex8.

colors = ['Red','Green','Write','Black']

print(f"First color is: {colors[0]}, last color is: {colors[-1]}")

Ex9.

examDate = (18,8,2022)

print("The examination will start from :%i/%i/%i"%examDate)

Ex10.

n = int(input("Nhap n: "))

nn = int( "%s%s" % (n,n) )

nnn = int( "%s%s%s" % (n,n,n) )

print(n+nn+nnn)

Ex11.

print(abs.\_\_doc\_\_)

Ex12.

import calendar

year = int(input("YEAR: "))

month = int(input("MONTH: "))

print(calendar.month(year, month))

Ex13.

print("""

a string that you "don't" have to escape

This

is a  ....... multi-line

heredoc string --------> example

""")

Ex14.

from datetime import date

firstDate = date(2022,8,18)

lastDate = date(2022,9,3)

result = lastDate - firstDate

print(result)

Ex15.

from math import pi

r = float(input("Nhap Ban kinh: "))

print(f"The volume of the sphereis: {4.0/3.0\*pi\*r\*\*3}")

Ex16.

def difference(n):

    return 17 - n if n <= 17 else (n - 17) \* 2

print(difference(30))

print(difference(2))

Ex17.

def near\_thousand(n):

      return ((abs(1000 - n) <= 100) or (abs(2000 - n) <= 100))

Ex18.

def sum\_thrice(x, y, z):

     sum = x + y + z

     if x == y == z:

      sum = sum \* 3

     return sum

Ex19.

def new\_string(str):

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

Ex20.

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))

Ex21.

number = int(input("Enter a number: "))

print("This is an odd number.") if number % 2 != 0 else print("This is an even number.")

Ex22.

def list\_count\_4(nums):

    count = 0

    for num in nums:

        if num == 4:

            count+=1

    return count

Ex23.

def substring\_copy(str, n):

    flen = 2

    result = ""

    if flen > len(str):

        flen = len(str)

    for i in range(n):

        result = result + str[:flen]

    return result

Ex24.

def isVowel(char):

    allVowels = 'aeiou'

    return char in allVowels

Ex25.

def is\_group\_member(group\_data, n):

    for value in group\_data:

        if n == value:

            return True

        return False

**Bài 2. Viết hàm thực hiện các chức năng sau:**

1. Tính:  
   a) (a + b),  
   b) a/b,  
   c) ab.
2. a = int(input("Nhap a: "))
3. b = int(input("Nhap b: "))
4. print(a+b)
5. print(a/b)
6. print(a\*\*b)

**2.** Tính diện tích hình chữ nhật khi biết bán kính

a = int(input("Nhap chieu dai: "))

b = int(input("Nhap chieu rong: "))

area = a \* b

print(f"Dien tich hinh chu nhat la: {area}")

**3.** Xuất tất cả các số nguyên tố trong 1 khoảng cho trước

from numpy import intp

from urllib3 import Retry

def isPrimeNum(i,o):

    if(o<2 or i < 2):

        return False

    for n in range(i,o+1):

        if o%n==0:

            print(n)

i = int(input("Nhap so nguyen duong i = "))

o = int(input("Nhap so nguyen duong n = "))

isPrimeNum(i,o)

**4.** Kiểm tra 1 số nguyên n có phải là số Fibonacci hay không

import math

n = int(input("Vui lòng nhập giá trị nguyên để kiểm tra số Fibonacci:"))

def check\_perfect\_square (m):

    n = int (math.sqrt (m))

    return n \* n == m

def check\_fibo (m ):

    return check\_perfect\_square (5 \* m \* m + 4) or check\_perfect\_square (5 \* m \* m - 4)

if (check\_fibo (n) == True):

    print (n, "là số Fibonacci")

else:

    print (n , "không phải là số Fibnacci")

**5.** Tìm số Fibonacci thứ n (dùng đệ quy và không đệ quy)

'''

Khong dung de quy

'''

def fibonacci\_without\_recursive(n):

    a=0;

    b=1;

    c=1;

    if(n<0):

        return -1

    else:

        for i in range(2,n):

            a = b

            b = c

            c = a + b

        return c

'''

Dung de quy

'''

def fibonacci\_with\_recursive(n):

    if(n<0):

        return -1

    elif (n==0 or n==1):

        return n

    else:

        return fibonacci\_with\_recursive(n-1) + fibonacci\_with\_recursive(n-2)

n = int(input("Nhap so nguyen duong n = "))

print(f"So fibonacci thu {n} khong dung de quy la: {str(fibonacci\_without\_recursive(n))}")

print(f"So fibonacci thu {n} dung de quy la: {str(fibonacci\_with\_recursive(n))}")

**6.** Tính tổng n số Fibonacci đầu tiên (dùng đệ quy và không đệ quy)

def fibonacci\_with\_recursive(n):

    if(n<0):

        return -1

    elif (n==0 or n==1):

        return n

    else:

        return fibonacci\_with\_recursive(n-1) + fibonacci\_with\_recursive(n-2)

list\_fibonacci = []

n = int(input("Nhập số nguyên dương n: "))

for i in range(0, n+1):

    list\_fibonacci.append(fibonacci\_with\_recursive(i))

print(f"Tổng của dãy {n} số fibonacci là {sum(list\_fibonacci)}")

7.Tính tổng căn bậc 2 của n số nguyên đầu tiên

def sumNFirstInteger(n):

    sum = 0

    for i in range(0,n+1):

        sum+=i\*\*2

    return sum

n = int(input("Nhap so nguyen duong n = "))

print(sumNFirstInteger(n))

8.Giải phương trình bậc 2: ax2 + bx + c=0

from math import sqrt

def ptBac2(a,b,c):

    x1 = 0

    x2 = 0

    delta = b\*\*2 - 4\*a\*c

    if delta < 0:

        print("Phương trình vô nghiệm")

    elif delta ==0:

        print(f"Phương trình có nghiệm kép x1=x2={-(b/(2\*a))}")

    else:

        x1= (-(b) + sqrt(delta))/(2\*a)

        x2 = (-(b) - sqrt(delta))/(2\*a)

        print(f"Phương trình có nghiệm phân biệt x1 = {x1}, x2 = {x2}")

a = float(input("Nhập hệ số a: "))

b = float(input("Nhập hệ số b: "))

while True:

    if a == 0 and b == 0:

        print("Một trong hai hệ số a, b phải khác 0: ")

        a = float(input("Nhập lại số a: "))

        b = float(input("Nhập lại số b: "))

    else:

        break

c = float(input("Nhập hệ số c: "))

ptBac2(a,b,c)

9.Tính n!

def giaiThua(n):

    if n==0:

        return 1

    return n \* giaiThua(n-1)

n = int(input("Nhập số nguyên dương n: "))

print(giaiThua(n))

10.In \* dạng tam giác dưới như hình bên, đầu vào là số hàng(cột)

def printTriangle(n):

    listStar = []

    for i in range(1,n+1):

        listStar.append("\*\t"\*i)

    print("\n".join(listStar))

n = int(input("Nhập số hàng: "))

printTriangle(n)

11.Đổi giờ - phút – giây: thời gian đầu vào là giây được đổi thành giờ, phút, giây.  
Xuất kết quả ra màn hình dưới dạng: giờ:phút:giây. Ví dụ: soGiay = 3770 thì xuất  
ra màn hình 1:2:50

def convertSec(s):

    sec = 0

    min = 0

    hour = 0

    if(s<60):

        sec = s

        print(f"{hour}h:{min}m:{sec}s")

    elif(s<3600 and s>=60):

        min = int((s-s%60)/60)

        sec =int(s% 60)

        print(f"{hour}h:{min}m:{sec}s")

    else:

        hour = int((s-s%3600)/3600)

        min = int(((s%3600) - (sec%3600)%60)/60)

        sec = int(s-min\*60-hour\*3600)

        print(f"{hour}h:{min}m:{sec}s")

convertSec(3700)

12.Cho một mảng số nguyên: (nên viết 2-3 cách)  
a) Xuât tất cả các số lẻ không chia hết cho 5  
b) Xuất tất cả các số Fibonacci  
c) Tìm số nguyên tố lớn nhất  
d) Tìm số Fibonacci bé nhất  
e) Tính trung bình các số lẻ  
f) Tính tích các phần tử là số lẻ không chia hết cho 3 trong mảng  
g) Đổi chỗ 2 phần tử của danh sách, đầu vào là 2 vị trí cần đổi chỗ  
h) Đảo ngược trật tự các phần tử của danh sách  
i) Xuất tất cả các số lớn thứ nhì của danh sách  
j) Tính tổng các chữ số của tất cả các số trong danh sách  
k) Đếm số lần xuất hiện của một số trong danh sách  
l) Xuất các số xuất hiện n lần trong danh sách  
m) Xuất các số xuất hiện nhiều lần nhất trong danh sách

import math

from os import system

list = [2,5,9,20,10,11,13,10,14,5,8,9,9]

def check\_perfect\_square (m):

    n = int (math.sqrt (m))

    return n \* n == m

def check\_fibo (m ):

    return check\_perfect\_square (5 \* m \* m + 4) or check\_perfect\_square (5 \* m \* m - 4)

def isOdd(i):

    return i and 1

def XuatSoLeKhongChiaHetCho5(list):

    sb=""

    for i in list:

        if i%5!=0:

            sb = sb + str(i) + ", "

    print(sb)

def XuatTatCaFibonacci(list):

    sb = ""

    for i in list:

        if check\_fibo(i) == True:

            sb = sb + str(i) + ","

    print(sb)

def TimSoNguyenToLonNhat(list):

    result = []

    for i in list:

        if i<2:

            return False

        if i%len(list)==0:

            result.append(i)

    print(max(result))

def TimSoFibonacciMin(list):

    result = []

    for i in list:

        if check\_fibo(i) == True:

            result.append(i)

    print(min(result))

def TrungBinhSoLe(list):

    result = []

    sum = 0

    for i in list:

        if isOdd(i) == 1:

            result.append(i)

    for i in result:

        sum+=i

    print(sum/len(result))

def TichSoLe(list):

    mul = 1

    for i in list:

        if (isOdd(i) == 1) and (i%3!=0):

            mul \*=i

    print(mul)

def Swap(list, x, y):

    list[x], list[y] = list[y],list[x]

    print(list)

def Rev(list):

    print(list[::-1])

def maxSecond(list):

    maxst = max(list[0], list[1])

    maxnd = max(list[0], list[1])

    for i in range(2, len(list)):

        if list[i] > maxst:

            maxnd = maxst

            maxst = list[i]

        elif (list[i]>maxnd) and (maxst != list[i]):

            maxnd=list[i]

    print(maxnd)

def totalDigitsOfNumber(n):

    total = 0

    while (n>0):

        total = total + n%10

        n = int(n/10)

    return total

def sumDigitsOfNumber(list):

    sum = 0

    for i in list:

        sum+= totalDigitsOfNumber(i)

    print(sum)

def countElement(list, n):

    if(n in list):

        print(list.count(n))

    else:

        print('Khong ton tai phan tu trong mang!')

def printAppearNElement(list, n):

    result = []

    for i in list:

        if list.count(i) == n:

            result.append(i)

    print(result)

def maxCountElement(list):

    b = []

    c = []

    for i in range(len(list)-1):

        b.append(list.count(list[i]))

    for i in range(len(b)-1):

        if b[i]== max(b):

            c.append(list[i])

    print(c[0])

while True:

    print('Chọn chức năng muốn thực hiện: ')

    print('Nhập 1: Xuât tất cả các số lẻ không chia hết cho 5')

    print('Nhập 2: Xuất tất cả các số Fibonacci')

    print('Nhập 3: Tìm số nguyên tố lớn nhất')

    print('Nhập 4: Tìm số Fibonacci bé nhất')

    print('Nhập 5: Tính trung bình các số lẻ')

    print('Nhập 6: Tính tích các phần tử là số lẻ không chia hết cho 3 trong mảng')

    print('Nhập 7: Đổi chỗ 2 phần tử của danh sách, đầu vào là 2 vị trí cần đổi chỗ')

    print('Nhập 8: Đảo ngược trật tự các phần tử của danh sách')

    print('Nhập 9: Xuất tất cả các số lớn thứ nhì của danh sách')

    print('Nhập 10: Tính tổng các chữ số của tất cả các số trong danh sách')

    print('Nhập 11: Đếm số lần xuất hiện của một số trong danh sách')

    print('Nhập 12: Xuất các số xuất hiện n lần trong danh sách')

    print('Nhập 13: Xuất các số xuất hiện nhiều lần nhất trong danh sách')

    print('Nhập 0: Thoát chương trình')

    try:

        action = int(input())

        if action == 0:

            break

        elif type(action) != int:

            print('XIN MỜI NHẬP LẠI')

            action = int(input())

            system("CLS")

            print('Chọn chức năng muốn thực hiện: ')

            print('Nhập 1: Xuât tất cả các số lẻ không chia hết cho 5')

            print('Nhập 2: Xuất tất cả các số Fibonacci')

            print('Nhập 3: Tìm số nguyên tố lớn nhất')

            print('Nhập 4: Tìm số Fibonacci bé nhất')

            print('Nhập 5: Tính trung bình các số lẻ')

            print('Nhập 6: Tính tích các phần tử là số lẻ không chia hết cho 3 trong mảng')

            print('Nhập 7: Đổi chỗ 2 phần tử của danh sách, đầu vào là 2 vị trí cần đổi chỗ')

            print('Nhập 8: Đảo ngược trật tự các phần tử của danh sách')

            print('Nhập 9: Xuất tất cả các số lớn thứ nhì của danh sách')

            print('Nhập 10: Tính tổng các chữ số của tất cả các số trong danh sách')

            print('Nhập 11: Đếm số lần xuất hiện của một số trong danh sách')

            print('Nhập 12: Xuất các số xuất hiện n lần trong danh sách')

            print('Nhập 13: Xuất các số xuất hiện nhiều lần nhất trong danh sách')

            print('Nhập 0: Thoát chương trình')

    except:

        print('CHÚNG TÔI HIỆN CHƯA PHÁT TRIỂN TÍNH NĂNG ĐÓ, XIN MỜI NHẬP LẠI')

        action = 0

    match action:

        case 1:

            system("CLS")

            print('===========================')

            print("Xuât tất cả các số lẻ không chia hết cho 5")

            print('===========================')

            XuatSoLeKhongChiaHetCho5(list)

            print('===========================')

        case 2:

            system("CLS")

            print('===========================')

            print("Xuất tất cả các số Fibonacci")

            print('===========================')

            XuatTatCaFibonacci(list)

            print('===========================')

        case 3:

            system("CLS")

            print('===========================')

            print("Tìm số nguyên tố lớn nhất")

            print('===========================')

            TimSoNguyenToLonNhat(list)

            print('===========================')

        case 4:

            system("CLS")

            print('===========================')

            print("Tìm số Fibonacci bé nhất")

            print('===========================')

            TimSoFibonacciMin(list)

            print('===========================')

        case 5:

            system("CLS")

            print('===========================')

            print("Tính trung bình các số lẻ")

            print('===========================')

            TrungBinhSoLe(list)

            print('===========================')

        case 6:

            system("CLS")

            print('===========================')

            print("Tính tích các phần tử là số lẻ không chia hết cho 3 trong mảng")

            print('===========================')

            TichSoLe(list)

            print('===========================')

        case 7:

            system("CLS")

            print('===========================')

            print("Đổi chỗ 2 phần tử của danh sách, đầu vào là 2 vị trí cần đổi chỗ")

            print('===========================')

            Swap(list,1,4)

            print('===========================')

        case 8:

            system("CLS")

            print('===========================')

            print("Đảo ngược trật tự các phần tử của danh sách")

            print('===========================')

            Rev(list)

            print('===========================')

        case 9:

            system("CLS")

            print('===========================')

            print("Xuất tất cả các số lớn thứ nhì của danh sách")

            print('===========================')

            maxSecond(list)

            print('===========================')

        case 10:

            system("CLS")

            print('===========================')

            print("Tính tổng các chữ số của tất cả các số trong danh sách")

            print('===========================')

            sumDigitsOfNumber(list)

            print('===========================')

        case 11:

            system("CLS")

            print('===========================')

            print("Đếm số lần xuất hiện của một số trong danh sách")

            print('===========================')

            countElement(list, 9)

            print('===========================')

        case 12:

            system("CLS")

            print('===========================')

            print("Xuất các số xuất hiện n lần trong danh sách")

            print('===========================')

            printAppearNElement(list,2)

            print('===========================')

        case 13:

            system("CLS")

            print('===========================')

            print("Xuất các số xuất hiện nhiều lần nhất trong danh sách")

            print('===========================')

            maxCountElement(list)

            print('===========================')