Assignment 1: 30 Days of Code: Python

NAME: AVANI NARVEKAR
ROLL NO: 42

Day 1: Data Types

1. **Declare variables: one of type int, one of type double, and one of type String.**
2. **Read lines of input from stdin (according to the sequence given in the Input Format section below) and initialize your variables.**
3. **Use the operator to perform the following operations:**
   A. **Print the sum of plus your int variable on a new line.**
   B. **Print the sum of plus your double variable to a scale of one decimal place on a new line.**
   C. **Concatenate with the string you read as input and print the result on a new line.**

In [5]:

```
i1 = 4
i2 = int(input('enter a integer:'))
```

```
enter a integer:3
```

In [6]:

```
print(i1+i2)
```

```
7
```

In [9]:

```
d1 = 15.3
d2= float(input('enter a float:'))
```

```
enter a float:3.6
```

In [10]:

```
print(d1+d2)
```

```
18.900000000000002
```

In [13]:

```
s1 = 'Hackerrank'
s2 = input('enter string:')
```

```
enter string: is the best
```

In [14]:

```
print(s1 + s2)
```

```
Hackerrank is the best
```

Day 2: Operators

Given the meal price (base cost of a meal), tip percent (the percentage of the meal price being added as tip), and tax percent (the percentage of the meal price being added as tax) for a meal, find and print the meal's total cost. Round the result to the nearest integer.

In [20]:

```
mealPrice= int(input('enter meal price:'))
tipPercent= int(input('enter tip percent:'))
taxPercent= int(input('enter tax percent:'))
totalPrice = mealPrice + (tipPercent*mealPrice + taxPercent*mealPrice)/100
print('total price: ', totalPrice)
```

```
enter meal price:100
enter tip percent:20
enter tax percent:8
total price:  128.0
```

Day 3: Intro to Conditional Statements

Given an integer, , perform the following conditional actions:

If n is odd, print Weird If n is even and in the inclusive range of 2 to 5, print Not Weird If n is even and in the inclusive range of 6 to 20, print Weird If n is even and greater than 20 , print Not Weird

In [22]:

```
n = int(input('enter a number:'))
if n%2 !=0:
    print('Weird')
elif 2<=n<=5:
    print('not weird')
elif 6<=n<=20:
    print('weird')
else:
    print('weird')
```

```
enter a number:7
Weird
```

Day4: Class vs Instance

Write a Person class with an instance variable, age, and a constructor that takes an integer, initialAge, as a parameter. The constructor must assign initialAge to age after confirming the argument passed as initalAge is not negative; if a negative argument is passed as , the constructor should set age to 0 and print Age is not valid, setting age to 0.. In addition, you must write the following instance methods:

1. **yearPasses() should increase the instance variable by 1.**
2. **amIOld() should perform the following conditional actions:**
   - **If age<13 , print You are young..**
   - **If age>= 13 and age<18, print You are a teenager..**
   - **Otherwise, print You are old..**

In [ ]:

```
class Person:
    def __init__(self,initialAge):
        # Add some more code to run some checks on initialAge
        self.age = initialAge
        if self.age < 0:
            print('Given age is invalid')
            self.age = 0

    def amIOld(self):
        # Do some computations in here and print out the correct statement to the console
        if self.age <13:
            print('You are young..')
        elif 13<=self.age<18:
            print('You are a teenager..')
        else:
            print('You are old..')
    def yearPasses(self):
        # Increment the age of the person in here
```

```
        self.age += 1

t = int(input())
for i in range(0, t):
    age = int(input())
    p = Person(age)
    p.amIOld()
    for j in range(0, 3):
        p.yearPasses()
    p.amIOld()
    print("")
```

```
4
15
You are a teenager..
You are old..

1
You are young..
You are young..

19
You are old..
You are old..
```

Day 5: Loops

Given an integer, n, print its first 10 multiples. Each multiple n x i (where 1<=i<=10) should be printed on a new line in the form: n x i = result.

In [29]:

```
num = int(input('enter the number '))
for i in range(1,11):
    print(num,'x',i,'=',num*i)
```

```
enter the number 4
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
4 x 9 = 36
4 x 10 = 40
```

Day 6: Let's Review

Given a string, S, of length N that is indexed from 0 to N-1, print its even-indexed and odd-indexed characters as 2 space-separated strings on a single line.

In [10]:

```
S = input('enter a string: ')
N = len(S)
```

```
enter a string: Hacker
```

In [11]:

```
for i in range(N):
    if i%2 == 0: print(S[i],'', end = "")
print(" ")
for i in range(N):
    if i%2 != 0: print(S[i],'',end="")
```

```
H c e
a k r
```

In [48]:

```
print(S[0::2] + " " + S[1::2]) #Slicing is used. syntax: StringVar[start:end:skip]
```

```
Hce akr
```

Day 7: Arrays

Given an array, A, of N integers, print A's elements in reverse order as a single line of space-separated numbers.

In [1]:

```
import math
import os
import random
import re
import sys
```

In [2]:

```
n = int(input("enter the number of elements: "))
```

```
enter the number of elements: 4
```

In [6]:

```
a = input("enter the array elements: ").rstrip().split()
```

```
enter the array elements: 1 2 3 4
```

In [9]:

```
for i in range(n):
    print(a[-1-i], end =" ")
```

```
4 3 2 1
```

Day 8: Dictionaries and Maps:

You are given a phone book that consists of your friend's names and their phone number. After that you will be given your friend's name as query. For each query, print the phone number of your friend.

In [18]:

```
n = int(input())
phoneDict = {}
```

```
2
```

In [19]:

```
for i in range(n):
    name, num = input().split()
    if len(num)==8 and name.islower() and num[0]!=0:
        phoneDict[name] = num
    else:
        print("name should be in lower-case and phone number should be 8 digit and not st
art with 0.")
```

```
avani 12345678
aditi 23456789
```

In [20]:

```
phoneDict
```

Out[20]:

```
{'avani': '12345678', 'aditi': '23456789'}
```

In [22]:

```
while True:
    query = input()
    if query in phoneDict:
        print(query, ":", phoneDict[query])
    elif query == "exit":
        break
    else:
        print("not found")
```

```
avani
avani : 12345678
ashu
not found
exit
```

Day 9: Recursion

Calculate the factorial of a number using recursion

In [17]:

```
n = int(input('enter a number: '))
def facto(n):
    if n == 1 or n == 0:
        return 1
    elif n<0:
        print("factorials does not exist")
    else:
        return n*facto(n-1)


print(facto(n))
```

```
enter a number: 3
6
```

Day 10: Binary Numbers

Given a base-10 integer, n, convert it to binary (base-2). Then find and print the base-10 integer denoting the maximum number of consecutive 1's in n's binary representation. When working with different bases, it is common to show the base as a subscript.

In [5]:

```
##n = int(raw_input().strip())
n = int(input().strip())

max_num = 0
count = 0
while n:
    while n&1:
        count += 1
        n>>=1
    max_num = max(count, max_num)
    if not n&1:
        count = 0
        n>>=1

def max(a,b):
    return a if a>b else b
```

```
print(max_num)
```

```
15
4
```

Day 11: 2D Arrays

Calculate the hourglass sum for every hourglass in A, then print the maximum hourglass sum.

In [41]:

```
arr = []
for arr_i in range(6):
    arr_temp = list(map(int,input().strip().split(' ')))
    arr.append(arr_temp)

for i in range(0,4):
    for j in range(0,4):
        max = 0
        sum = arr[i][j]+arr[i][j+1]+arr[i][j+2]+arr[i+1][j+1]+arr[i+2][j]+arr[i+2][j+1]
+arr[i+2][j+2]
        if i == 0 and j ==0:
            max = sum
        if sum > max:
            max = sum
print(max)
```

```
1 1 1 0 0 0
0 1 0 0 0 0
1 1 1 0 0 0
0 0 0 2 4 4
0 0 0 0 2 0
0 0 0 1 2 4
19
```

Day 12: Inheritance

https://www.hackerrank.com/challenges/30-inheritance/problem

In [43]:

```
class Person():
    def __init__(self, firstName, lastName, idNumber):
        self.firstName = firstName
        self.lastName = lastName
        self.idNumber = idNumber

    def printPerson(self):
        print("Name: ", self.firstName + ",", self.lastName)
        print("ID: ", self.idNumber)

class Student(Person):
    def __init__(self, firstName, lastName, idNumber, scores):
        super().__init__(firstName, lastName, idNumber)
        self.scores = scores

    def average(self):
        total = 0.0
        for score in self.scores:
            total += score

        avg = total/len(scores)
        if 90 <= avg <= 100:
            return 'O'
        elif 80 <= avg < 90:
            return 'E'
        elif 70 <= avg < 80:
            return 'A'
```

```
        elif 55 <= avg < 70:
            return 'P'
        elif 40 <= avg < 55:
            return 'D'
        return 'T'


line = input().split()
firstName = line[0]
lastName = line[1]
idNumber = line[2]
scores = list(map(int, input().split()))
s = Student(firstName, lastName, idNumber, scores)
s.printPerson()
print("Grade: ", s.average())
```

```
avani narvekar 123456789
79 89 48
Name:  avani, narvekar
ID:  123456789
Grade:  A
```

Day 13: Abstract Classes

https://www.hackerrank.com/challenges/30-abstract-classes/problem

In [45]:

```
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author

class myBook(Book):
    def __init__(self, title, author, price):
        super().__init__(title,author)
        self.price = price
    def display(self):
        print('Title:',self.title)
        print('Author:', self.author)
        print('Price:', self.price)

title = input("enter the title: ")
author = input("enter the author: ")
price = input("enter the price: ")
new_book = myBook(title,author,price)
new_book.display()
```

```
enter the title: KAFKA ON THE SHORE
enter the author: HARUKI MURAKAMI
enter the price: 499
Title: KAFKA ON THE SHORE
Author: HARUKI MURAKAMI
Price: 499
```

Day 14: Scope

https://www.hackerrank.com/challenges/30-scope/problem

In [87]:

```
class differ:
    def __init__(self,a):
        self.__elements = a

    def computeDiff(self):
        maximum = 0
```

```
        for i in range(len(self.__elements)):
            for j in range(len(self.__elements)):
                absolute = abs(self.__elements[i] - self.__elements[j])
                if absolute>maximum:
                    maximum = absolute
        self.maximumDifference = maximum


_ = input()
a = [int(e) for e in input().split(" ")]
d= differ(a)
d.computeDiff()
print(d.maximumDifference)
```

```
4
1 3 5 7
6
```

Day 15: Linked List

https://www.hackerrank.com/challenges/30-linked-list/problem

In [90]:

```
class Node:
    def __init__(self,data):
        self.data = data
        self.next = None

class Solution:
    def display(self,head):
        current = head
        while current:
            print(current.data, end = ' ')
            current = current.next

    def insert(self,head,data):
        if head is None:
            head = Node(data)
        else:
            curr = head
            while curr.next:
                curr = curr.next
            curr.next = Node(data)
        return head

mylist = Solution()
T = int(input())
head = None
for i in range(T):
    data = int(input())
    head = mylist.insert(head,data)
mylist.display(head)
```

```
3
4
5
2
4 5 2
```

Day 16: Exceptions - String to Integer

https://www.hackerrank.com/challenges/30-exceptions-string-to-integer/problem

In [68]:

```
try:
    a = int(input())
```

```
    print(a)
except:
    print("Bad string")
```

```
a
Bad string
```

Day 17: More Exceptions

https://www.hackerrank.com/challenges/30-more-exceptions/problem

In [76]:

```
class Calculator(Exception):
    def power(self,n,p):
        if n<0 or p<0:
            raise Exception("n and p should be non-negative")
        else:
            return pow(n,p)
mycal = Calculator()
t = int(input("enter the numbers: "))
for i in range(t):
    n,p = map(int,input().split())
    try:
        ans = mycal.power(n,p)
        print(ans)
    except Exception as e:
        print(e)
```

```
enter the numbers: 4
3 5
243
2 5
32
-1 9
n and p should be non-negative
6 2
36
```

Day 18: Queues and Stacks

https://www.hackerrank.com/challenges/30-queues-stacks/problem

In [80]:

```
class Solution:
    def __init__(self):
        self.stack = []
        self.queue = []

    def push_char(self, ch):
        self.stack.append(ch)

    def enqueue_char(self, ch):
        self.queue.append(ch)

    def pop_char(self):
        try:
            x = self.stack[-1]
            self.stack = self.stack[:-1]
            return x
        except:
            return None

    def dequeue(self):
        try:
            x = self.queue[0]
            self.queue = self.queue[1:]
```

```
            return x
        except:
            return None

str_s = Solution
S = 'racecar'

for c in S:
    str_s.push_char(c)
    str_s.enqueue_char(c)

flag = True
while True:
    from_stack = str_s.pop_char()
    from_queue = str_s.dequeue()
    if from_stack == None:
        break
    if from_stack!=from_queue:
        flag = False
        break
if flag:
    print("Palindrome")
else:
    print("not a palindrome")
```

```
Palindrome
```

Day 19: Interfaces

https://www.hackerrank.com/challenges/30-interfaces/problem

In [91]:

```
class AdvancedArithmetic(object):
    def divisorSum(n):
        raise NotImplementedError

class Calculator(AdvancedArithmetic):
    def divisorSum(self, n):
        sum_ = 0
        for i in range(1, n + 1):
            if (n % i == 0):
                sum_ += i
        return sum_


s = Calculator()
s.divisorSum(13)
```

Out[91]:

```
14
```

Day 20: Sorting

In [95]:

```
import sys
n = int(input())
my_array = [int(x) for x in input().split(" ")]
no_swaps = 0
for i in range(0,n):
    for j in range(0,n-1):
        if my_array[j]>my_array[j+1]:
            temp = my_array[j]
            my_array[j] = my_array[j+1]
            my_array[j+1] = temp
            no_swaps +=1
```

```
    if no_swaps == 0:
        break
print("array is sorted in",str(no_swaps),'steps')
print("first element: ", str(my_array[0]))
print("last element: ", str(my_array[len(my_array)-1]))
```

```
5
1 3 5 4 2
array is sorted in 4 steps
first element:  1
last element:  5
```

Day 22: BST

In [114]:

```
class Node:
    def __init__(self,data):
        self.right=self.left=None
        self.data=data

class Solution:
    def insert(self, root, data):
        if root==None:
            return Node(data)
        else:
            if data<=root.data:
                cur = self.insert(root.left,data)
                root.left = cur
            else:
                cur = self.insert(root.right,data)
                root.right = cur
                return root

    def getHeight(self,root):
        if root == None or root.left == None and root.right == None:
            return 0
        else:
            return 1 + max(self.getHeight(root.left),self.getHeight(root.right))

T = int(input("Size:"))
myTree = Solution()
root = None
for i in range(T):
    data = int(input())
    root = myTree.insert(root,data)
height = myTree.getHeight(root)
print("Height: ",height)
```

```
Size:6
1
2
3
4
5
6
Height: 5
```

Day 23: BST Level order traversal

In [112]:

```
class Node:
    def __init__(self,data):
        self.right=self.left=None
        self.data = data

class Solution:
```

```
    def insert(self,root,data):
        if root==None:
            return Node(data)
        else:
            if data<=root.data:
                cur=self.insert(root.left,data)
                root.left=cur
            else:
                cur=self.insert(root.right,data)
                root.right=cur
                return root

    def getHeight(self,root):
        if root == None or root.left == None and root.right == None:
            return 0
        else:
            return 1 + max(self.getHeight(root.left),self.getHeight(root.right))


    def levelOrder(self, root):
        ret = ""
        queue = [root]
        while queue:
            current = queue.pop(0)
            ret += str(current.data) + " "
            if current.left:
                queue.append(current.left)
            if current.right:
                queue.append(current.right)
        print(ret[:-1])

T=int(input("Size:"))
myTree=Solution()
root = None
for i in range(T):
    data=int(input())
    root=myTree.insert(root,data)
myTree.levelOrder(root)
```

```
Size:5
1
2
3
4
5
1 2 3 4 5
```

Day 24: Linked list

In [67]:

```
class Node:
    def __init__(self,data):
        self.data = data
        self.next = None


class Solution:
    def insert(self,head,data):
        p = Node(data)
        if head==None:
            head=p
        elif head.next==None:
            head.next=p
        else:
            start=head
            while(start.next!=None):
                start=start.next
            start.next=p
        return head
```

```
    def display(self,head):
        current = head
        while current:
            print(current.data,end=' ')
            current = current.next

    def removeDuplicates(self,head):
        node = head
        while node:
            if node.next:
                if node.data == node.next.data:
                    node.next = node.next.next
                    continue
            node = node.next
        return head

mylist= Solution()
T=int(input("Size: "))
head = None
for i in range(T):
    data=int(input())
    head=mylist.insert(head,data)

print("\nBefore Removing Duplicates: ")
mylist.display(head)
head=mylist.removeDuplicates(head)
print("\nAfter Removing Duplicates")
mylist.display(head)
```

```
Size: 6
1
2
3
4
4
5

Before Removing Duplicates:
1 2 3 4 4 5
After Removing Duplicates
1 2 3 4 5
```

Day 25: Running time and complexity

In [64]:

```
def prime(i):
    for j in range(3,int((i*0.5)+1),2):
        if i%j == 0:
            return False
    return True

x = [int(input()) for i in range(int(input()))]
for i in x:
    if(i==1)or i>2 and (i%2==0):
        print(i, "is not prime")
    elif prime(i):
        print(i,"is prime")
    else:
        print(i,"is not prime")
```

```
3
1
9
7
1 is not prime
9 is not prime
7 is prime
```

Day 26: Nested Logic

In [55]:

```
returned_date = list(map(int,input("enter the return date: ").split(" ")))
expected_date = list(map(int,input("enter the expected date: ").split(" ")))
fine = 0
if returned_date[2]>expected_date[2]:
    fine = 10000
elif returned_date[2] == expected_date[2]:
    if returned_date[1]>expected_date[1]:
        fine = (returned_date[1]-expected_date[1])*500
    elif returned_date[1]==expected_date[1]:
        if returned_date[0]>expected_date[0]:
            fine = (returned_date[0]-expected_date[0])*15
print("fine: ",fine)
```

```
enter the return date: 12 12 20
enter the expected date: 01 12 20
fine:  165
```

Day 27: Testing

In [58]:

```
def minimum_index(seq):
    if len(seq)==0:
        raise ValueError("Cannot get the minimum value index")
    min_idx=0
    for i in range(1,len(seq)):
        if seq[i]<seq[min_idx]:
            min_idx=i
    return min_idx

class Emptyarray():
    @staticmethod
    def get_array():
        return list()

class Uniquevalues():
    @staticmethod
    def get_array():
        return [5, 2, 8, 3, 1, -6, 9]
    @staticmethod
    def get_exp_result():
        return 5

class ExactlyTwoDiff():
    @staticmethod
    def get_array():
        return [5, 2,8,31,-6,9,-6,19]
    @staticmethod
    def get_exp_result():
        return 5

def WithEmptyarray():
    try:
        seq=Emptyarray.get_array()
        res=min_index(seq)
    except ValueError as e:
        pass
    else :
        assert False

def WithUniqueval():
    seq=Uniquevalues.get_array()
    assert len(seq)>=2

    assert len(list(set(seq)))==len(seq)
```

```
    exp_result=Uniquevalues.get_exp_result()
    res=min_index(seq)
    assert res==exp_result

def WithExactlyTwodiff():
    seq=ExactlyTwoDiff.get_array()
    assert len(seq)>=2
    tmp=sorted(seq)
    assert tmp[0]==tmp[1] and (len(tmp)==2 or tmp[1]< tmp[2])
    exp_result=ExactlyTwoDiff.get_exp_result()
    res=min_index(seq)
    assert result==exp_result

Emptyarray()
Uniquevalues()
ExactlyTwoDiff()
print("OK!")
```

```
OK!
```

Day 28: Regex, Pattern, and Intro to Databases

In [52]:

```
A=[]
for i in range(int(input())):
    name, email = map(str,input().split())
    if email.endswith('@gmail.com'):
        A.append(name)

print("Names: ")
for i in sorted(A):
    print(i)
```

```
3
avani avani@gmail.com
aditi aditi@gmail.com
ashu ashu@gmail.com
Names:
aditi
ashu
avani
```

Day 29: Bitwise AND

In [49]:

```
for i in range(int(input())):
    n,k = map(int,input().split())
    print("value: ",k-1 if ((k-1)|k)<=n else k-2)
```

```
2
2 3
value:  1
3 4
value:  2
```