ML Assignment 1 : Hackerank 30 days of code

Aryaman Gautam

J001

Day 0 : Hello, World.

To complete this challenge, you must save a line of input from stdin to a variable, print Hello, World. on a single line, and finally, print the value of your variable on a second line.

In [8]:
```
a = input("Please enter what you'd like to display : \n")

print("\n\n\nHello, World.")
print(a)
```

```
Please enter what you'd like to display :
Nice to meet you.



Hello, World.
Nice to meet you.
```

Day 1 : Data Types

Declare 3 variables one of type int, one of type double, and one of type String. Read 3 lines of input from stdin (according to the sequence given in the Input Format section below) and initialize your 3 variables.

Use the + operator to perform the following operations: Print the sum of i plus your int variable on a new line. Print the sum of d plus your double variable to a scale of one decimal place on a new line. Concatenate s with the string you read as input and print the result on a new line.

In [7]:

```
#i,d,s
redo = 1

while redo == 1 :
    try:
        i = int(input("Please enter an integer : "))
        print()
        break
    except:
        print("\n!Please input only an integer value.!\n")

while redo == 1 :
    try:
        d = float(input("Please enter a decimal : "))
        print()
        break
    except:
        print("\n!Please input only a decimal/numeric value.!\n")

while redo == 1 :
    try:
        s = input("Please input a string : ")
        print()
        break
    except:
        print("\n!Please input only a string.!\n")



# int would convert 34.3 to 34 but trying to convert "34.3" gives an error



my_int_var = 4
my_double_var = 34.897
my_string = " This was added string"

lis = [i,my_int_var,i+my_int_var,d,my_double_var,d+my_double_var,s,my_string,s+my_string]
```

```
op = "\n\n\n\nSum (int) of {} & {} is : {} \nSum (dec) of {} & {} is : {} \nConcatination of '{}' & '{}' is : {}"
print(op.format(*lis))

#when using list in format use *
```

```
Please enter an integer : 34.567

!Please input only an integer value.!

Please enter an integer : 34

Please enter a decimal : 456f

!Please input only a decimal/numeric value.!

Please enter a decimal : 457.893

Please input a string : This was inputted. !
```

```
Sum (int) of 34 & 4 is : 38
Sum (dec) of 457.893 & 34.897 is : 492.78999999999996
Concatination of 'This was inputted. !' & ' This was added string' is : This was inputted. ! This was added string
```

Day 2 : Operators

Given the meal price (base cost of a meal), tip per cent (the percentage of the meal price being added as a tip), and tax per cent (the percentage of the meal price being added as tax) for a meal, find and print the meal's total cost.

In [1]:

```
meal_price = float(input("Please enter your meal price : "))
tip_percent = float(input("What percent of your bill would you like to tip : "))
tax_percent = float(input("Please enter the tax rate : "))

tip = meal_price*tip_percent/100
tax = meal_price*tax_percent/100

# follows BODMAS 40/100*12 = 4.8
```

```
print("\n\n\nYour meal cost : {} \nYour Tipping amount : {} \nAmount Taxxed : {} \nFinal bill : {}".format(meal_price
```

```
Please enter your meal price : 450
What percent of your bill would you like to tip : 10
Please enter the tax rate : 5



Your meal cost : 450.0
Your Tipping amount : 45.0
Amount Taxxed : 22.5
Final bill : 517.5
```

Day 3 : Intro to Conditional Statements

According to Hackerrank, you should follow these four given conditions.

- If ‘n‘ is odd, print Weird.
- If ‘n‘ is even and in the inclusive range of 2 to 5, print Not Weird.
- ‘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 [9]:

```
for i in range(10):

    n = input("Please enter a number : ")
    a = n.split(".")

    siz = len(a)

    if siz >2:
        print("Not a propper numerical input")
        break
    else:
        chk = a[-1]
        chk = int(chk) #for checking if even or odd
        n = float(n)

    if chk%2 != 0:
        print("Wierd")
```

```
    else:
        if n >= 2 and n <= 5:
            print("Not weird")
        elif n >= 6.0 and n <= 20.0:
            print("weird")
        elif n > 20.0 :
            print("Not weird")
        else:
            print("even and (less than 2 or b/w 5&6.)")

    print()
```

```
Please enter a number : -5
Wierd

Please enter a number : 5
Wierd

Please enter a number : 3.4
Not weird

Please enter a number : 4
Not weird

Please enter a number : 12.898
weird

Please enter a number : 24
Not weird

Please enter a number : 24.56896
Not weird

Please enter a number : -2
even and (less than 2 or b/w 5&6.)

Please enter a number : 5.4
even and (less than 2 or b/w 5&6.)

Please enter a number : 16
weird
```

- #issue with using float
-
- a = float("24.4")
- print(a)
- print(type(a))
- print(a%2) #issue here
- if a % 2 != 0: print("hiya")
-
-
-
-
-
-
-
-
- 24.4
- <class 'float'>
- 0.3999999999999986
- hiya

Day 4 : 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 initialAge is not negative;
if a negative argument is passed as initialAge, 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:

yearPasses() should increase the age instance variable by 1.
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 [14]:

```
class Person:
    def __init__(self,aged):
        if aged >= 0:
            self.age = aged
        else:
            print("Age is not valid, setting age to 0.")
            self.age = 0

    def yearPasses(self):
        self.age += 1
        print("Now age is : ",self.age)

    def amIOld(self):
        if self.age <13:
            print("You are young.")
        elif self.age <18:
            print("You are a teenager")
        else:
            print("You are old")

ag = int(input("Enter age"))
print()
P = Person(ag)
P.amIOld()


print("\n\n\n")


ag = int(input("Enter age"))
print()
P = Person(ag)
P.amIOld()
```

```
print("\n\n\n")

i = 1
while i <= 6:
    P.yearPasses()
    i += 1

print("\n\n\n")

P.amIOld()

print("\n\n\n")

i = 1
while i <= 6:
    P.yearPasses()
    i += 1
print()
P.amIOld()
```

```
Enter age-4

Age is not valid, setting age to 0.
You are young.



Enter age8

You are young.



Now age is :  9
Now age is :  10
Now age is :  11
Now age is :  12
Now age is :  13
Now age is :  14
```

```
You are a teenager



Now age is :  15
Now age is :  16
Now age is :  17
Now age is :  18
Now age is :  19
Now age is :  20

You are old
```

Day 5 : Loops

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

In [15]:

```
i = 1
while i == 1:
    try:
        a = int(input("Please enter an integer : "))
        break
    except:
        print("Please enter only an integer to proceed")

while i <= 10:
    print(a,"*",i," = ",a*i)
    i += 1
```

```
Please enter an integer : 7
7 * 1  =  7
7 * 2  =  14
7 * 3  =  21
7 * 4  =  28
7 * 5  =  35
7 * 6  =  42
7 * 7  =  49
7 * 8  =  56
```

```
7 * 9  =  63
7 * 10  =  70
```

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 (see the Sample below for more detail).

In [18]:
```
S = input("Enter a string : ")
N = len(S)
even = ""
odd = ""
for i in range(N):
    if i%2 == 0:
        even = even + S[i]
    else:
        odd = odd + S[i]

#string.replace(" ", "") if o/p was for not considering spaces

print()
print(even,"  ",odd)
```

```
Enter a string : Hello it's me

Hloi' e    el tsm
```

Day 7 : Arrays

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

In [25]:
```
import numpy as np


a = int(input("Please enter the number of elements you wish for the array to have : "))
print()

A = np.random.randint(0,a*10,a)
print(A)
```

```
print()

print(A[::-1])

print(np.flip(A))     # A.reverse() -> is for lists
```

```
Please enter the number of elements you wish for the array to have : 5

[35  0  7 11  0]

[ 0 11  7  0 35]
[ 0 11  7  0 35]
```

Day 8 : Dictionaries and maps

Given n names and phone numbers, assemble a phone book that maps friends' names to their respective phone numbers.
You will then be given an unknown number of names to query your phone book for.
For each name queried, print the associated entry from your phone book on a new line in the form name=phoneNumber;
if an entry for the name is not found, print Not found instead.

In [60]:

```
n = int(input("number of ppl you wish to store in the phonebook : "))
print()

dictn = {}
def PhoneBook(name,phno):
    dictn[name] = phno

for i in range(n):
    entry = input("Entry {} : ".format(i+1))
    entry = entry.strip()
    entry = entry.split()
    PhoneBook(entry[0],entry[1])

print()
print()
print(dictn)
print()
print()
```

```
n = int(input("Enter Number of users you wish to find : "))
for i in range(n):
    usr = input("user : ")
    usr = usr.strip()
    print(dictn.get(usr,"Not Found ! "))
```

```
number of ppl you wish to store in the phonebook : 5

Entry 1 : A 4780602474
Entry 2 : B 8532953966
Entry 3 : C 5685384287
Entry 4 : D 9280191222
Entry 5 : E 7844365934


{'A': '4780602474', 'B': '8532953966', 'C': '5685384287', 'D': '9280191222', 'E': '7844365934'}


Enter Number of users you wish to find : 2
user : B
8532953966
user : G
Not Found !
```

Day 9 : Recursion

we need to develop a program that takes an integer input and then prints the factorial of that integer input on the output screen

In [62]:

```
intg = int(input("Enter integer for which you'd like to find factorial : "))

def fact(n):
    if n == 1:
        return 1
    else:
        return n*fact(n-1)

print(fact(intg))
```

```
Enter integer for which you'd like to find factorial : 5
120
```

Day 10 : Binary Numbers

we need to develop a program that can accept integer as an input and then convert it into a binary number
and then into in base 10 integer.
we need to print the base 10 integer that denotes the maximum number of consecutive 1's in the binary representation of the input.

In [99]:

```
no = int(input("Enter integer to convert to binary : "))
print()
print()

rmd = ""
tmp = no
while no > 0:
    rm = no % 2
    no = no//2
    rmd = rmd + str(rm)
rmd = rmd[::-1]
print("Binary of {}\u2081\u2080 is : {}\u2082".format(tmp,rmd))
print()
print()

rmd = rmd.replace("0"," ")
def func(a):
    return len(a)
a = map(func,rmd.split())
print("No. of 1's that occur consecutively in binary are : ",max(list(a)))
```

```
Enter integer to convert to binary : 142


Binary of 142₁₀ is : 10001110₂


No. of 1's that occur consecutively in binary are :  3
```

Day 11 : 2D arrays

we need to develop a program that can take a 2d array as an input and then print the maximum hourglass sum of that array

In [9]:
```
arr = []
for arr_i in range(6):
   arr_temp = list(map(int,input().strip().split(' ')))
   arr.append(arr_temp)
max = 0

for i in range(0,4):
    for j in range(0,4):
        sum = 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 2 4 4 0
0 0 0 2 0 0
0 0 1 2 4 0
19
```

In [118…
```
sumr = 0
if m % 2 == 0:
    d1 = list(range(0,(m-1)//2))
    d2 = d1[::-1]
    for i in d1:
        sumr = sum(arr)
else:
    d1 = list(range())
    d2 = d1[::-1]



for j in range(0,m): #say 6 so 0,1,2,3,4,5
    if j <= (m-1)//2:
        condn = j*-1
    else:
        condn = j
```

```
    for k in range(,):
```

```
11
[[4 3 4]
 [4 5 6]]
(2, 3)
[4, 3, 2, 1, 0]
```

Day 12 : Inheritance

we have two classes, Person and Student. where the Person is the base class and the Student is the derived class. we need to inherit all the data from Person class and use them in Student class.

In [129…

```
class Person:
    def __init__(self, fname, lname,idd):
        self.fn = fname
        self.ln = lname
        self.ided = idd

    def randasin(self):
        self.clss =  np.random.randint(0,5,1)[0]


class Student(Person):
    def __init__(self, fname, lname,idd,scr):
        self.score = scr
        Person.__init__(self, fname, lname, idd)
    def details(self):
        Person.randasin(self)
        print("Name:    ",self.fn)
        print("Surname: ",self.ln)
        print("ID:      ",self.ided)
        print("Class:   ",self.clss)
        print("Score:   ",self.score)

obj = Student("U","O",786475,99)
obj.details()
```

```
Name:     U
```

```
Surname:  0
ID:       786475
Class:    3
Score:    99
```

Day 13 : Abstract Classes

we have two classes Book and Solution. we need to make a new class MyBook that inherited from Book and can print the details.

In [130…

```python
class Book:

    def __init__(self,title,author):
        self.title = title
        self.author = author


class MyBook(Book):

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

    def display(self):
        print("Title:", self.title)
        print("Author:", self.author)
        print("Price:", self.price)

obj = MyBook("Harry Potter","J.k Rowling",894.98)
obj.display()
```

```
Title: Harry Potter
Author: J.k Rowling
Price: 894.98
```

Day 14 : Scope

we have given A class constructor that takes an array of integers as a parameter and saves it to the instance variable. A computeDifference method finds the maximum absolute difference between any numbers and stores it in the instance variable.

In [137…

```
import random

class MaxDiff:
    def __init__(self,arr):
        self.arry = arr
        self.diff
    def diff(self):
        self.i1 = random.randint(1,(len(self.arry)-1))
        self.i2 = self.i1 - 1
        print(self.i1,self.i2)
        self.diff = self.arry[self.i1] -  self.arry[self.i2]
        print(self.diff)

obj = MaxDiff([2,12,4,9,6,14])
obj.diff()
```

```
5 4
8
```

Day 15 : Linked List

insert function in your editor so that it creates a new Node (pass data as the Node constructor argument) and inserts it at the tail of the linked list referenced by the head parameter. Once the new node is added, return the reference to the head node.

In [140…

```
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):
        new = Node(data)
        if head:
            tail = head
            while tail.next:
```

```
                tail = tail.next
            tail.next = new
            return head
        else:
            return new

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

```
5
2
3
4
5
6
2 3 4 5 6
```

Day 16

we need to develop a program that can take a string as an input and then can convert it into the equivalent integer.

In [141…

```
try:
    ip = int(input("Enter an integer"))
    print(ip)
except :
    print("Please enter an integer only")
```

```
Enter an integer   45
45
```

Day 17

Write a Calculator class with a single method: int power(int,int). The power method takes two integers, n and p, as parameters and returns the integer result of n^p. If either n or p is negative, then the method must throw an exception with the message: n and p should be non-negative

In [144…

```
class Calculator:
    def power(self,n,p):
        try:
            if n|p < 0 :
                print("n & p both should be positive")
                raise ValueError
            else:
                return n**p
        except ValueError:
            print("ValueError Exception!")



obj = Calculator()
print(obj.power(2,3))
print(obj.power(-1,4))
```

```
8
n & p both should be positive
ValueError Exception!
None
```

In [ ]:

```
#Day 18 onward chk
```

Day 18

Write the following declarations and implementations:

Two instance variables: one for your stack, and one for your queue. A void pushCharacter(char ch) method that pushes a character onto a stack. A void enqueueCharacter(char ch) method that enqueues a character in the queue instance variable. A char popCharacter() method that pops and returns the character at the top of the stack instance variable.

In [145…

```
class Solution:
    def __init__(self):
        self.mystack = list()
        self.myqueue = list()
        return(None)
```

```
    def pushCharacter(self, char):
        self.mystack.append(char)

    def popCharacter(self):
        return(self.mystack.pop(-1))

    def enqueueCharacter(self, char):
        self.myqueue.append(char)

    def dequeueCharacter(self):
        return(self.myqueue.pop(0))

# read the string s
s=input()
#Create the Solution class object
obj=Solution()

l=len(s)
# push/enqueue all the characters of string s to stack
for i in range(l):
    obj.pushCharacter(s[i])
    obj.enqueueCharacter(s[i])

isPalindrome=True
'''
pop the top character from stack
dequeue the first character from queue
compare both the characters
'''
for i in range(l // 2):
    if obj.popCharacter()!=obj.dequeueCharacter():
        isPalindrome=False
        break
#finally print whether string s is palindrome or not.
if isPalindrome:
    print("The word, "+s+", is a palindrome.")
else:
    print("The word, "+s+", is not a palindrome.")
```

```
hello
The word, hello, is not a palindrome.
```

Day 19 :

we need to take an integer input and then we need to subtract that integer and then add all the numbers and print the sum of them on the output screen.

In [157…

```
class Calculator():
    def divisorSum(self, n):
        #pass
        x = []
        for i in range(1, n+1):
            if n % i == 0:
                x.append(int(i))
            else:
                pass
        print(x)
        sum = 0
        for j in x:
            sum = sum + j
        return sum

n = int(input())
my_calculator = Calculator()
s = my_calculator.divisorSum(n)
print(s)
```

```
12
[1, 2, 3, 4, 6, 12]
28
```

Day 20 :

Given an array, a, of size n distinct elements, sort the array in ascending order using the Bubble Sort algorithm above. Once sorted, print the following 3 lines:

- No. of swaps :
- 1st element
- Last element

In [163…

```
def bubbleSort(arr):
    n = len(arr)
    cnt = 0
    # Traverse through all array elements
    for i in range(n):

        # Last i elements are already in place
        for j in range(0, n-i-1):

            # traverse the array from 0 to n-i-1
            # Swap if the element found is greater
            # than the next element
            if arr[j] > arr[j+1] :
                arr[j], arr[j+1] = arr[j+1], arr[j]
                cnt = cnt + 1
    return arr , cnt

# Driver code to test above
arr = list(map(int,input("Enter array : ").strip().split()))

arr , count = bubbleSort(arr)
print(arr)
print()
print("No. of Swaps : ",count)
print("1st element",arr[0])
print("Last element",arr[-1])
```

```
Enter array : 9 3 6 2 13 2
[2, 2, 3, 6, 9, 13]

No. of Swaps :  9
1st element 2
Last element 13
```

Day 21 :

we need to write a function printArray that can take an array of generic elements as a parameter and that function can print each element of its generic array parameter on a new line

In [164…

```
def printArray(arr):
```

```
    for i in arr:
        print(i)

arr = list(map(int,input("Enter array : ").strip().split()))
printArray(arr)
```

```
Enter array : 12 45 23 4 5 6 7
12
45
23
4
5
6
7
```

Day 22:

we need to complete a function getHeight that can take a pointer input and then print the height of the binary search tree.

In [165…

```
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:
            leftDepth = self.getHeight(root.left)
            rightDepth = self.getHeight(root.right)
```

```
            if leftDepth > rightDepth:
                return leftDepth + 1
            else:
                return rightDepth + 1
        else:
            return -1

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

```
9
12
4
3
7
8
45
4
67
34
3
```

Day 23:

A level-order traversal, also known as a breadth-first search, visits each level of a tree's nodes from left to right, top to bottom. You are given a pointer, root, pointing to the root of a binary search tree.

In [167…

```
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

    from collections import deque
    def levelOrder(self,root):

        queue = self.deque([root]) if root else self.deque()

        while queue:
            node = queue.popleft()
            print(node.data, end=' ')

            if node.left: queue.append(node.left)
            if node.right: queue.append(node.right)

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

```
6
4
6
5
3
8
2
4 3 6 2 5 8
```

Day 24 :

we need to complete the function removeDuplicates that can delete any node in the linked list that is duplicate. and return the head of the updated list.

In [169…

```
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):
        #Write your code here
        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())
head=None
for i in range(T):
    data=int(input())
    head=mylist.insert(head,data)
head=mylist.removeDuplicates(head)
mylist.display(head)
```

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

Day 25 :

we need to develop a program that can take integer input and then print that whether a number is a prime number or not a prime number.

In [172…

```
for _ in range(int(input())):
    num = int(input())
    if(num == 1):
        print("Not prime")
    else:
        if(num % 2 == 0 and num > 2):
            print("Not prime")
        else:
            for i in range(3, int(num**(1/2))+1, 2):
                if num % i == 0:
                    print("Not prime")
                    break
            else:
                print("Prime")
```

```
3
12
Not prime
3
Prime
```

```
5
Prime
```

Day 26 :

Your local library needs your help! Given the expected and actual return dates for a library book, create a program that calculates the fine (if any). The fee structure is as follows:

If the book is returned on or before the expected return date, no fine will be charged (i.e. fine=0) . If the book is returned after the expected return day but still within the same calendar month and year as the expected return date, fine= 15 Hackos x number of days late. If the book is returned after the expected return month but still within the same calendar year as the expected return date, the fine= 500Hackos x number of days late. If the book is returned after the calendar year in which it was expected, there is a fixed fine of 10000 Hackos

In [174…

```
returned_date = list(map(int,input().split(' ')))
expected_date = list(map(int,input().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)
```

```
31 8 2001
15 1 2001
3500
```

Day 27 :

For a given array of n integers, the function returns the index of the element with the minimum value in the array. If there is more than one element with the minimum value, the returned index should be the smallest one. If an empty array is passed to the function, it should raise an Exception.

In [173…

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

class TestDataEmptyArray():

    @staticmethod
    def get_array():
        return list()

class TestDataUniqueValues():

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

    @staticmethod
    def get_expected_result():
        return 5

class TestDataExactlyTwoDifferentMinimums():

    @staticmethod
    def get_array():
        return [5, 2, 8, 3, 1, -6, 9, -6, 10]

    @staticmethod
    def get_expected_result():
        return 5


def TestWithEmptyArray():
    try:
        seq = TestDataEmptyArray.get_array()
        result = minimum_index(seq)
    except ValueError as e:
        pass
```

```
    else:
        assert False


def TestWithUniqueValues():
    seq = TestDataUniqueValues.get_array()
    assert len(seq) >= 2

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

    expected_result = TestDataUniqueValues.get_expected_result()
    result = minimum_index(seq)
    assert result == expected_result


def TestiWithExactyTwoDifferentMinimums():
    seq = TestDataExactlyTwoDifferentMinimums.get_array()
    assert len(seq) >= 2
    tmp = sorted(seq)
    assert tmp[0] == tmp[1] and (len(tmp) == 2 or tmp[1] < tmp[2])

    expected_result = TestDataExactlyTwoDifferentMinimums.get_expected_result()
    result = minimum_index(seq)
    assert result == expected_result

TestWithEmptyArray()
TestWithUniqueValues()
TestiWithExactyTwoDifferentMinimums()
print("OK")
```

```
OK
```

Day 28 :

Consider a database table, Emails, which has the attributes First Name and Email ID. Given N rows of data simulating the Emails table, print an alphabetically-ordered list of people whose email address ends in @gmail.com.

In [4]:

```
import re

if __name__ == '__main__':
```

```
    N = int(input())

    pattern = r"@gmail\.com$"
    regex = re.compile(pattern)
    firstNames = []

    for N_itr in range(N):
        firstNameEmailID = input().split()

        firstName = firstNameEmailID[0]

        emailID = firstNameEmailID[1]

        if regex.search(emailID):
            firstNames.append(firstName)

    firstNames.sort()

    for name in firstNames:
        print(name)
```

```
4
calp calp@yahoo.com
josh josh@gmail.com
hen hen@farm.com
mouse barn@gmail.com
josh
mouse
```

Day 29 :

we have given a set S in which we need to find two integers A and B. such that the value of A and B is maximum possible and also less than a given integer K.

In [3]:

```
def FindMaxAB(n, k):
    max_ab = 0
    for i in range(k - 2, n):
        for j in range(i + 1, n + 1):
            ab = i & j
            if ab == k - 1:
                return ab
```

```
            if max_ab < ab < k:
                max_ab = ab
    return max_ab


for i in range(int(input().strip())):
    n, k = map(int, input().split())
    print(FindMaxAB(n, k))
```

```
3
2 5
0
8 5
4
12 7
6
```