# 30 Days of Code (Python)

## Day 0 - Hello World

```
# Read a full line of input from stdin and save it to our dynamically typed variable, inpu
input_string = input()

# Print a string literal saying "Hello, World." to stdout.
print('Hello, World.')
print(input_string)

# TODO: Write a line of code here that prints the contents of input_string to stdout.
```

## Day 1 - Data types

```
i = 4
d = 4.0
s = 'HackerRank '
# Declare second integer, double, and String variables.

# Read and save an integer, double, and String to your variables.
my_int= int(input())
my_double= float(input())
my_Str=input()

# Print the sum of both integer variables on a new line.
print(my_int+i)
# Print the sum of the double variables on a new line.
print(my_double+d)
# Concatenate and print the String variables on a new line
print(s+my_Str)
# The 's' variable above should be printed first.
```

## Day 2 - Operators

```
def solve(meal_cost, tip_percent, tax_percent):
    total_tip= (tip_percent * meal_cost)/100
    total_tax= (tip_percent * meal_cost)/100

    print(total_tip, total_tax)
    return round(total_tip+ total_tax+ meal_cost)

if __name__ == '__main__':
```

```
    meal_cost = float(input().strip())

    tip_percent = int(input().strip())

    tax_percent = int(input().strip())

    print(solve(meal_cost, tip_percent, tax_percent))
```

## Day 3 - Intro to conditional statements

```
def compare(N):
    if N%2 == 0:
        if 2<=N<=5:
            return 'Not Weird'
        elif 6<=N<=20:
            return 'Weird'
        else:
            return 'Not Weird'
    else:
        return 'Weird'


if __name__ == '__main__':
    N = int(input().strip())

    print(compare(N))
```

## Day 4 - Class vs Instance

```
class Person():

    def __init__(self,intialAge):

        if intialAge>0:
            self.age= intialAge
        else:
            self.age=0
            print('Age is not valid, setting age to 0.')

    def yearPasses(self):
         self.age= self.age +1

    def amIOld(self):

        if self.age< 13:
            print('You are young.')
        elif 13<=self.age<18:
            print('You are a teenager.')
        else:
            print('You are old.')
```

```
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("")
```

## Day 5 - Loops

```
def print_multiples(n):

    for i in range(1,11):
        print(f'{n} x {i} = {n*i}')


if __name__ == '__main__':
    n = int(input().strip())

    print_multiples(n)
```

## Day 6 - Lets Review

```
def spaced_string(str):

    even= ''
    odd=''
    for i in range(0,len(str)):

        if i%2==0:
            even= even + str[i]
        else:
            odd= odd + str[i]

    return(f'{even} {odd}')

if __name__== '__main__':

    T= int(input().strip())
    str_arr=[]
    for i in range(0,T):
        S= input().strip()
        str_arr.append(S)

    for i in str_arr:
        print(spaced_string(i))
```

```
print(spaced_string(1))
```

## Day 7 - Arrays

```
def Reverse(lst):

    reverse_str= str(lst[-1])
    for i in reversed(lst[:-1]):
        reverse_str=reverse_str +' '+ str(i)
    return reverse_str.rstrip()

if __name__ == '__main__':
    n = int(input().strip())

    arr = list(map(int, input().rstrip().split()))

    print(Reverse(arr))
```

## Day 8 - Dictonaries and Maps

```
n = int(input())
name_numbers = [input().split() for _ in range(n)]
phone_book = {k: v for k,v in name_numbers}
while True:
    try:
        name = input()
        if name in phone_book:
            print('%s=%s' % (name, phone_book[name]))
        else:
            print('Not found')
    except:
        break
```

## Day 9 - Recursion

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

if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    n = int(input().strip())

    result = factorial(n)
```

```
    fptr.write(str(result) + '\n')

    fptr.close()
```

## Day 10 - Binary numbers

```
#!/bin/python3

def convert_binary(num):
    bin=[]
    consec=0
    arr=[]


    #converting to binary
    while int(num/2>0):

        bin.append(int(num%2))
        num=int(num/2)
    bin.append(0)

    for i in range(0,len(bin)):
        if bin[i]==1:
            consec=consec+1
        elif bin[i]==0:
            arr.append(consec)
            consec=0

    return max(arr)

if __name__ == '__main__':
    n = int(input().strip())

    print(convert_binary(n))
```

## Day 11 - 2D Arrays

```
res = []

for x in range(0, 4):
    for y in range(0, 4):
        s = sum(arr[x][y:y+3]) + arr[x+1][y+1] + sum(arr[x+2][y:y+3])
        res.append(s)

print(max(res))
```

## Day 12 - Inheritance

```
class Person:
  def __init__(self, firstName, lastName, idNumber):
    self.firstName = firstName
    self.lastName = lastName
    self.idNumber = idNumber
  def printPerson(self):
    print("Name:", self.lastName + ",", self.firstName)
    print("ID:", self.idNumber)

class Student(Person):

    def __init__(self,first,last,id,scores):
        super().__init__(first, last,id)
        self.scores=scores

    def calculate(self):

        #average score
        average= sum(self.scores)/len(self.scores)

        #grades
        if 90<=average<=100:
            return 'O'
        elif 80<=average<90:
            return 'E'
        elif 70<=average<80:
            return 'A'
        elif 55<=average<70:
            return 'P'
        elif 40<=average<55:
            return 'D'
        else:
            return 'T'



line = input().split()
firstName = line[0]
lastName = line[1]
idNum = line[2]
numScores = int(input()) # not needed for Python
scores = list( map(int, input().split()) )
s = Student(firstName, lastName, idNum, scores)
s.printPerson()
print("Grade:", s.calculate())
```

## Day 13 - Abstract classes

```
from abc import ABCMeta, abstractmethod
class Book(object, metaclass=ABCMeta):
    def __init__(self,title,author):
        self.title=title
        self.author=author
    @abstractmethod
    def display(): pass

#Write MyBook class
class MyBook(Book):

    def __init__(self, title, author, price):
        super().__init__(title, author)
        self.price = price
    def display(self):
        print(f'Title: {self.title}')
        print(f'Author: {self.author}')
        print(f'Price: {self.price}')



title=input()
author=input()
price=int(input())
new_novel=MyBook(title,author,price)
new_novel.display()
```

## Day 14 - Scope

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

  # Add your code here
    maximumDifference=0


    def computeDifference(self):
        self.__elements.sort()
        self.maximumDifference=self.__elements[-1]-self.__elements[0]


# End of Difference class

_ = input()
a = [int(e) for e in input().split(' ')]

d = Difference(a)
d.computeDifference()

print(d.maximumDifference)
```

## Day 15 - Linked Lists

```
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):
    #Complete this method
        newNode=Node(data)
        if head == None:
            head=newNode
            return head
        else:
            current=head
            while current.next != None:
                current=current.next
            current.next=newNode
            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);
```

## Day 16 - Exceptions-string to integer

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


if __name__ == '__main__':
    S = input()

try:
    S=int(S)
    print(S)
except:
    print('Bad String')
```

```
        print('Bad String')
```

## Day 17 - More exceptions

```
#Write your code here
class Calculator():
    def power(self,n,p):
        if n>=0 and p>=0:
            return pow(n,p)
        else:
            return  "n and p should be non-negative"


myCalculator=Calculator()
T=int(input())
for i in range(T):
    n,p = map(int, input().split())
    try:
        ans=myCalculator.power(n,p)
        print(ans)
    except Exception as e:
        print(e)
```

## Day 18 - Queues and Stack

```
import sys
class Solution:
    # Write your code here
    def __init__(self):
        self.stack = []
        self.queue = []

    def pushCharacter(self, x):
        self.stack.append(x)

    def popCharacter(self):
        return  self.stack.pop()

    def enqueueCharacter(self, x):
        self.queue.insert(0, x)

    def dequeueCharacter(self):
        return self.queue.pop()




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

```
#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.")
```

## Day 19 - Interfaces

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

class Calculator(AdvancedArithmetic):
    def divisorSum(self, n):
        divisor=[]

        for i in range(1,n+1):
            if n%i==0:
                divisor.append(i)
        return sum(divisor)


n = int(input())
my_calculator = Calculator()
s = my_calculator.divisorSum(n)
print("I implemented: " + type(my_calculator).__bases__[0].__name__)
print(s)
```

## Day 20 - Bubble sort

```
if __name__ == '__main__':
```

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

    a = list(map(int, input().rstrip().split()))

    #number of tracks swapped during a single array traversal
    numberofSwaps=0

    for i in range(0,len(a)-1):

        for j in range(0,len(a)-1):
            if a[j]>a[j+1]:
                a[j+1], a[j] = a[j], a[j+1]
                numberofSwaps= numberofSwaps + 1

        #if no elements swapped during traversal, array is sorted
        if numberofSwaps==0:
            break

    print(f'Array is sorted in {numberofSwaps} swaps.')
    print(f'First Element: {a[0]}')
    print(f'Last Element: {a[-1]}')
```

**Day 21 not avaliable in python**

## Day 22 - Binary Search Trees

```
class node:
  def getHeight(self,root):
        if root is None or (root.left is None and root.right is None):
            return 0
        else:
            return max(self.getHeight(root.left),self.getHeight(root.right))+1
T = int(input())
```

## Day 23 - BST level order traversal

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

## Day 24 - More Linked lists

```
def removeDuplicates(self,head):
    #Write your code here
    current = head
    while (current.next):
        if (current.data == current.next.data):
            current.next = current.next.next
        else:
            current = current.next

    return head
```

## Day 25 - Running time and complexity

```
import math
n=int(input())
l=[]

def checkPrime(x):
    c=0
    if x==1:
        print("Not prime")
        return
    for i in range(1,round(math.sqrt(x))+1,2):
        if i==1:
            if x%2==0:
                print("Not prime")
                c+=1
                return
        elif x%i==0:
            print("Not prime")
            c+=1
            return
    print("Prime")


for i in range(0,n):
    l.append(int(input()))

for a in l:
    checkPrime(a)
```

## Day 26 - Nested logic

```
rd, rm, ry = [int(x) for x in input().split(' ')]
ed, em, ey = [int(x) for x in input().split(' ')]

if (ry, rm, rd) <= (ey, em, ed):
    print(0)
elif (ry, rm) == (ey, em):
    print(15 * (rd - ed))
elif ry == ey:
    print(500 * (rm - em))
else:
    print(10000)
```

## Day 27 - Testing

```
class TestDataEmptyArray:
    @staticmethod
    def get_array():
        return []

class TestDataUniqueValues:
    @staticmethod
    def get_array():
        return [1, 2,3]

    @staticmethod
    def get_expected_result():
        return [1,2,3].index(min([1, 2,3]))


class TestDataExactlyTwoDifferentMinimums:
    @staticmethod
    def get_array():
        return [1, 2, 3, 1]

    @staticmethod
    def get_expected_result():
        return [1, 2, 3, 1].index(min([1, 2, 3, 1]))
```

## Day 28 - Regex, Patterns and Intro to Databases

```
import re

def regex_checker(email):
    return re.search('\w+@gmail.com',email)

if __name__ == '__main__':
    N = int(input().strip())
    names=[]
```

```
    for N_itr in range(N):
        first_multiple_input = input().rstrip().split()

        firstName = first_multiple_input[0]

        emailID = first_multiple_input[1]
        if regex_checker(emailID):
            names.append(firstName)

    names.sort()
    for i in names:
        print(i)
```

## Day 29 - Bitwise AND

```
#!/bin/python3

import os


def bitwiseAnd(n, k):
    bits=[]
    s=[i+1 for i in range(n)]
    for i in range(0,len(s)):
        for j in range(i+1,len(s)):
            bits.append(s[i]&s[j])

    highest=bits[0]
    for i in range(1,len(bits)):
        if highest<bits[i] and bits[i]<k:
            highest= bits[i]

    return highest

if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    t = int(input().strip())

    for t_itr in range(t):
        first_multiple_input = input().rstrip().split()

        count = int(first_multiple_input[0])

        lim = int(first_multiple_input[1])

        res = bitwiseAnd(count, lim)

        fptr.write(str(res) + '\n')

    fptr.close()
```