In [2]:

```
#DAY 0
inputString=input()
print("hello world")
print(inputString)
```

```
my name is siddhanth
hello world
my name is siddhanth
```

In [6]:

```
#DAY 1
i = 2
d = 2.0
str = 'abc'
x = 3
y = 4.0
z = "input"
print(i+x)
print(d+y)
print(str+z)
```

```
5
6.0
abcinput
```

In [12]:

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

In [ ]:

```

```

In [26]:

```
#DAY 2
mealcost = float(input())
tax = int(input())
tip = int(input())
tax = tax*mealcost/100;
tip = tip*mealcost/100;
total = mealcost + tip + tax;
x = int(math.floor(total))

y = "{}".format(x)
print("total mealcost is" + y + "rupees")
```

```
12
20
8
total mealcost is15rupees
```

In [1]:

```
#DAY 3
n = int(input().strip())
if n%2==1:
    ans = "Weird"

elif n>20:
    ans = "Not Weird"
```

```
elif n>=6:
    ans = "Weird"

else:
    ans = "Not Weird"

print(ans)
```

```
5
Weird
```

In [6]:

```
#day4
class Person:
    def __init__(self,initialAge):
        if(initialAge > 0):
            self.age = initialAge
        else:
            print("Age is not valid, setting age to 0.")
            self.age = 0
    def amIOld(self):

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

    def yearPasses(self):

        self.age += 1
age=int(input('enter your age'))
p = Person(age)
p.amIOld()
for j in range(0,3):
    p.yearPasses()
p.amIOld()
print("")
```

```
enter your age12
You are young.
You are a teenager.
```

In [8]:

```
#day 5
n=int(input("Enter a number for its multiples"))
for i in range(1,11):
    print(f"{n} * {i} = {n*i}")
```

```
Enter a number for its multiples3
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
3 * 4 = 12
3 * 5 = 15
3 * 6 = 18
3 * 7 = 21
3 * 8 = 24
3 * 9 = 27
3 * 10 = 30
```

In [9]:

```
#day 6
inputString = input("Enter a string ")
even = ""
odd = ""
```

```
for i in range(0, len(inputString)):
    if i % 2 == 0:
        even = even + inputString[i]
    else:
        odd = odd + inputString[i]

print(even)
print(odd)
```

```
Enter a string hello
hlo
el
```

In [10]:

```
#day 7
arr = []
lim = int(input("Enter the number of integers to be put in the array: "))
for i in range(0,lim):
    num = int(input(f"Integer {i+1}: "))
    arr.append(num)
for i in reversed(range(0,lim)):
    print(arr[i])
```

```
Enter the number of integers to be put in the array: 2
Integer 1: 1
Integer 2: 3
3
1
```

In [16]:

```
#day 9
def factorial(n):
    if n == 0:
        return 1

    return n * factorial(n-1)

n = int(input("Enter a number to calc the factorial: "))
factorial(n)
```

```
Enter a number to calc the factorial: 6
```

Out[16]:

```
720
```

In [18]:

```
#day 10
num = int(input("Enter a number in decimal system: "))
n = num
count = 0
while n >= 1:
    r = n % 2
    n = n//2
    if r == 1:
        count += 1
    else:
        count = 0
print("\n")
if count == 1:
    print("0 ones are together")
else:
    print(f"{count} ones are together")
```

```
Enter a number in decimal system: 54


2 ones are together
```

In [20]:

```
#day 12
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,firstName,lastName,idNumber,scores):
        super().__init__(firstName,lastName,idNumber)
        self.studentSC = scores

    def calculate(self):
        sum = 0
        for mark in self.studentSC:
            sum += mark
        avg = sum/len(self.studentSC)
        if avg >= 90:
            grade = 's'
        elif avg >= 80:
            grade = 'a'
        elif avg >= 70:
            grade = 'b'
        elif avg >= 55:
            grade = 'c'
        elif avg >= 40:
            grade = 'D'
        else:
            grade = 'e'

        return grade



fname = input("Enter your first name: ")
lname = input("Enter your last name: ")
studentId = input("Enter your student ID")
print("Enter your scores for PCM out of 100: ")
scores = []
for i in range(0,3):
    subject = int(input(f"Subject {i + 1}: "))
    scores.append(subject)

p = Person(fname,lname,studentId)


s = Student(fname, lname, studentId, scores)
s.printPerson()
print("Grade:", s.calculate())
```

```
Enter your first name: siddhanth
Enter your last name: Nair
Enter your student ID123
Enter your scores for PCM out of 100:
Subject 1: 80
Subject 2: 90
Subject 3: 100
Name: Nair, siddhanth
ID: 123
Grade: s
```

In [21]:

```
#day 13
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

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

    def display(self):
        print("Title: %s \nAuthor: %s\nPrice: %s" % (title, author, price))

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

```
no longer human
osamu dazai
500
Title: no longer human
Author: osamu dazai
Price: 500
```

In [22]:

```
#Day 14
class Difference:
    def __init__(self, a):
        self.__elements = a
    def computeDifference(self):
        self.maximumDifference = max(self.__elements) - min(self.__elements)
        return None

d = Difference(a = [1,2,5])
d.computeDifference()
print(d.maximumDifference)
```

```
4
```

In [23]:

```
#day 15
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)
        elif head.next is None:
            head.next = Node(data)
        else:
            self.insert(head.next, 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)
```

```
4
2
6
3
2
2632
```

In [24]:

```
#Day 16
import sys

s = input().strip()
try:
    r = int(s)
    print(r)
except ValueError:
    print("Bad String")
```

```
stardenbardenharden
Bad String
```

In [29]:

```
#Day 18
from collections import deque

class Solution:
    def __init__(self):
        self.stack = deque()
        self.queue = deque()

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

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

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

    def dequeueCharacter(self):
        return self.queue.popleft();

s=input()
obj=Solution()
l=len(s)

for i in range(l):
    obj.pushCharacter(s[i])
    obj.enqueueCharacter(s[i])
isPalindrome=True

for i in range(l//2):
    if obj.popCharacter() != obj.dequeueCharacter():
        isPalindrome = False
        break

if isPalindrome:
    print("The word, "+s+", is a palindrome.")
else:
    print("The word, "+s+", is not a palindrome.")
```

```
malayalam
The word, malayalam, is a palindrome.
```

In [30]:

```
#day 194
class AdvArithmetic(object):
    def divisorSum(n):
        raise NotImplementedError

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

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

```
4
7
```

In [33]:

```
#day 20
if __name__ == '__main__':
    n=int(input().strip())
    a=list(map(int,input().rstrip().split(' ')))
    numberOfSwaps = 0
    for i in range(0,n):
        for j in range(0,n-1):
            if(a[j]>a[j+1]):
                temp=a[j]
                a[j]=a[j+1]
                a[j+1]=temp
                numberOfSwaps +=1
        if(numberOfSwaps == 0):
            break
print("Array is sorted in "+str(numberOfSwaps)+" swaps.")
print("First Element: "+str(a[0]))
print("Last Element: "+str(a[n-1]))
```

```
4
8 2 4 3
Array is sorted in 4 swaps.
First Element: 2
Last Element: 8
```

In [34]:

```
#day 22
def getHeight(self,root):
    if not root:
        return -1
    else:
        a = self.getHeight(root.left)
        b = self.getHeight(root.right)
        if (a > b):
            return (a + 1)
        else:
            return (b + 1)
    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
```

In [35]:

```
#Day 23
import sys
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 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])

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

```
5
2
4
1
7
4
2 1 4 4 7
```

In [ ]:

```
#day 24
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):
        current=head
        while(current.next):
            if(current.data==current.next.data):
                current.next=current.next.next
```

```
            else:
                current=current.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);
```

In [ ]:

```
#day 25
import math
def check_prime(num):
    if num == 1:
        return "Not prime"
    sq = int(math.sqrt(num))
    for x in range(2,sq+1):
        if num % x == 0:
            return "Not prime"
    return "Prime"
t = int(input())
for i in range(t):
    number=int(input())
    print(check_prime(number))
```

In [55]:

```
#day 26
return_date = [int(i)for i in input().split()]
due_date = [int(i) for i in input().split()]
if return_date[2] > due_date[2]:
    print(10000)
else:
    if return_date[2] == due_date[2]:
        if return_date[1] > due_date[1]:
            print(500 * (return_date[1] - due_date[1]))
        elif return_date[1] == due_date[1] and return_date[0] > due_date[0]:
            print(15 * (return_date[0] - due_date[0]))
        else:
            print(0)
    else:
        print(0)
```

```
2 12 2001
12 2 2004
0
```

In [57]:

```
#day 27
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(object):


    def get_array():
        return []
```

```
class TestDataUniqueValues(object):


    def get_array():
        return [7, 4, 3, 8, 14]


    def get_expected_result():
        return 2

class TestDataExactlyTwoDifferentMinimums(object):


    def get_array():
        return [7, 4, 3, 8, 3, 14]


    def get_expected_result():
        return 2
```

In [ ]:

```
#day 28

if __name__ == '__main__':
    N = int(input().strip())
    names = []
for a0 in range(N):
    firstName,emailID = input().rstrip().split(' ')
    firstName,emailID = [str(firstName),str(emailID)]
    match = re.search(r'[\w\.-]+@gmail.com',emailID)
    if match:
        names.append(firstName)
names.sort()
for name in names:
    print(name)
```

In [ ]:

```
#day 29

import sys


t = int(input().strip())
for a0 in range(t):
    n, k = input().strip().split(' ')
    n, k = [int(n), int(k)]
    print(k-1 if ((k-1) | k) <= n else k-2)
```

```
5
5 6
4
7 8
6
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: