Machine Learning Assignment

Aditya Gavankar (J072)

Exp 1 :- Hackerrank 30 days of Python code

Day 0: Hello World

In [1]:

```
print('Hello, World!')

inputString = input()
print(inputString)
```

```
Hello, World!
Welcome to the Hackerrank World!!
Welcome to the Hackerrank World!!
```

Day 1: Data Types

In [4]:

```
i, d, s = 10, 8.5, "Awesome "

j = int(input("Enter integer:"))
e = float(input("Enter double:"))
t = input("Enter string:")

print(i + j)
print(d + e)
print(s + t)
```

```
Enter integer:5
Enter double:7.5
Enter string:Photo
15
16.0
Awesome Photo
```

Day 2: Operators

In [6]:

```
mealCost = float(input("Meal price: "))
tip = int(input("Tip: "))
tax = int(input("Taxes: "))

tip = tip * mealCost / 100;
tax = tax * mealCost / 100;
totalcost = mealCost + tip + tax;

print ("The total meal cost is %s dollars." %str(int(round(totalcost, 0))))
```

```
Meal price:5
Tip:7
Taxes:9
The total meal cost is 6 dollars.
```

Day 3: Conditionals

In [7]:

```
import sys

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)
```

```
7
Weird
```

Day 4: Classes vs Inheritance

In [13]:

```
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


for t in range(3):
    initialAge=int(input('Enter Age: '))
    person=Person(initialAge = initialAge)
    person.amIOld()
    person.yearPasses()
```

```
Enter Age: 9
```

```
You are young.
Enter Age: 15
You are a teenager.
Enter Age: 19
You are old.
```

Day 5: Loops

In [14]:

```
import sys


N = int(input().strip())
for i in range(1, 11):
    print(str(N) +" x " + str(i) + " = " + str(N*i))
```

```
10
10 x 1 = 10
10 x 2 = 20
10 x 3 = 30
10 x 4 = 40
10 x 5 = 50
10 x 6 = 60
10 x 7 = 70
10 x 8 = 80
10 x 9 = 90
10 x 10 = 100
```

Day 6: Review

In [3]:

```
import sys

def printEvenIndexChar(s):
    l = len(s)
    output = ""
    for i in range(0, l, 2):
        output += s[i]
    return output

def printOddIndexChar(s):
    l = len(s)
    output = ""
    for i in range(1, l, 2):
        output += s[i]
    return output

t = int(input())
for a0 in range(0,t):
    s = input()
    print(printEvenIndexChar(s) + " " + printOddIndexChar(s))
```

```
1
Satoshi
Stsi aoh
```

Day 7: Arrays

In [7]:

```
import sys


n = int(input().strip())
arr = list(map(int,input().strip().split(' ')))
ans = ""
for i in range(len(arr)-1 , -1, -1):
    ans += str(arr[i]) + " "

print(ans)
```

```
3
8 5
5 8
```

Day 8: Maps and Dictionaries

In [1]:

```
dict={}

for i in range(2):
    name=input()
    phone=int(input())
    dict[name]=[]
    dict[name].append(phone)

print(dict)
key_list=list(dict.keys())
val_list=list(dict.values())
position=key_list.index(input())
print(val_list[position])
```

```
Aditya
1
Amit
2
{'Aditya': [1], 'Amit': [2]}
Aditya
[1]
```

Day 9: Recursion

In [20]:

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

n = int(input())
print(factorial(n))
```

```
10
3628800
```

Day 10: Binary Numbers

In [2]:

```
import sys

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

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

print(max_num)
```

```
1010
6
```

Day 11: 2D Arrays/Matrix

In [8]:

```
import sys


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] + a
r[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 2 3 4 5 6
7 8 9 1 2 3
4 5 6 7 8 9
9 8 7 6 5 4
3 2 1 9 8 7
6 5 4 3 2 1
53
```

Day 12: Inheritance

In [10]:

```
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, fName, lName, sId, scores):
        super().__init__(fName, lName, sId)
        self.scores = scores

    def calculate(self):
        avg = 0.0
        for score in self.scores:
            avg += score

        avg = avg/len(self.scores)
        if avg < 40:
            return 'T'
        elif avg < 55:
            return 'D'
        elif avg < 70:
            return 'P'
        elif avg < 80:
            return 'A'
        elif avg < 90:
            return 'E'
        else:
            return 'O'


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())
```

```
Aditya Gavankar 910235
2
75 87
Name: Gavankar, Aditya
ID: 910235
Grade: E
```

Day 13: Abstract Classes

In [4]:

```
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: %s\nAuthor: %s\nPrice: Rs %s" %(self.title, self.author,
self.price))


my_book1 = MyBook('The Diary Of A Young Girl','Anne Frank',1000)
my_book1.display()
```

```
Title: The Diary Of A Young Girl
Author: Anne Frank
Price: Rs 1000
```

Day 14: Scope

In [5]:

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

    def maximum_difference(self):
        return max(self.elements)-min(self.elements)

diff=Difference([1, 2, 3, 4, 5, 6, 7, 8, 9])
diff.maximum_difference()
```

Out[5]:

```
8
```

Day 15: Linked Lists

In [7]:

```
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)
```

```
5
4
3
2
1
0
4 3 2 1 0
```

Day 16: Handling Exceptions

In [29]:

```
import sys


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

```
Broken
Bad String
```

Day 17: Throwing Exceptions

In [17]:

```
class Calculator(Exception):
    def power(self, n, p):
        if (n < 0 or p < 0):
            raise Calculator("n and p should be non-negative")
        else:
            return pow(n, p)

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)
```

```
3
2 6
64
-1 3
n and p should be non-negative
```

n and p should be non negative
3 4
81

Day 18: Stacks and Queues

In [20]:

```
import sys
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)
# 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

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

madam
The word, madam, is a palindrome.

Day 19: Interfaces

In [18]:

```
class AdvancedArithmetic(object):
```

```
    def divisorSum(n):
        raise NotImplementedError

class Calculator(AdvancedArithmetic):
    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)
```

```
10
18
```

Day 20: Sorting

In [19]:

```
import sys


n = int(input().strip())
a = list(map(int, input().strip().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]) )
```

```
5
7 10 3 5 1
Array is sorted in 8 swaps.
First Element: 1
Last Element: 10
```

Day 21: Generics

Unavailable in Python

Day 22: Binary Search Tree

In [8]:

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

height = myTree.getHeight(root)
print("Height of tree: ",height)
```

```
8
4
3
2
6
7
1
8
9
Height of tree:  4
```

Day 23: Level Order Transversal

In [9]:

```
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)
```

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

Day 24: Linked Lists 2

In [21]:

```
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)
```

```
7
3
2
5
4
4
6
1
3 2 5 4 6 1
```

Day 25: Time/Complexity

In [13]:

```
import math

def check_prime(num):
    if num is 1:
        return "Not prime"
    sq = int(math.sqrt(num))
    for x in range(2, sq+1):
        if num % x is 0:
            return "Not prime"
    return "Prime"

t = int(input())
for i in range(t):
    number = int(input())
    print(check_prime(number))
```

```
<>:4: SyntaxWarning: "is" with a literal. Did you mean "=="?
<>:8: SyntaxWarning: "is" with a literal. Did you mean "=="?
<>:4: SyntaxWarning: "is" with a literal. Did you mean "=="?
<>:8: SyntaxWarning: "is" with a literal. Did you mean "=="?
<ipython-input-13-00828938cc21>:4: SyntaxWarning: "is" with a literal. Did you mea
n "=="?
  if num is 1:
<ipython-input-13-00828938cc21>:8: SyntaxWarning: "is" with a literal. Did you mea
n "=="?
  if num % x is 0:
```

```
3
```

```
13
Prime
15
Not prime
21
Not prime
```

Day 26: Nested Logic

In [17]:

```
da, ma, ya = input().split(' ')
da = int(da)
ma = int(ma)
ya = int(ya)
de, me, ye = input().split(' ')
de = int(de)
me = int(me)
ye = int(ye)
fine = 0
if(ye==ya):
    if(me < ma):
        fine = (ma - me) * 500

    elif((me == ma) and (de < da)):
        fine = (da - de) * 15

elif(ye < ya):
    fine = 10000

print(fine)
```

```
11 12 2021
25 7 2021
2500
```

Day 27: Testing

In [ ]:

```
def minimum_index(seq):
    if len(seq) == 0:
        raise ValueError("Cannot get the minimum value index from an empty sequenc
e")
    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):

    @staticmethod
    def get_array():
        return []

class TestDataUniqueValues(object):

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

```
    @staticmethod
    def get_expected_result():
        return 2

class TestDataExactlyTwoDifferentMinimums(object):

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

    @staticmethod
    def get_expected_result():
        return 2
```

Day 28: Regex

In [11]:

```
import sys
import re

N = int(input().strip())
names = []
for a0 in range(N):
    firstName,emailID = input().strip().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 )
```

```
2
aditya aditya@gmail.com
uk uk@gmail.com
aditya
uk
```

Day 29: Bitwise AND

In [12]:

```
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)
```

```
3
5 3
2
9 6
5
4 4
2
```

In [ ]: