PYTHON İLE

TEKNİK MÜLAKAT SORULARI

ULUDAG IEEE COMPUTER SOCIETY

Kolay Sorular

SORU 1

Virgülle ayrılmış sözcük dizisini girdi olarak kabul eden ve sözcükleri alfabetik olarak sıraladıktan sonra çizgi ile ayrılmış bir dizide yazdıran bir program yazınız.

Programa aşağıdaki girdinin sağlandığını varsayalım:

"okul, merhaba, durak, kitap"

Ardından, çıktı şöyle olmalıdır:

durak - kitap - merhaba - okul

ÇÖZÜM 1.1

```
items = []
#ilk bölme işlemini yapıyoruz.
for x in content.split(','):
    items.append(x)

#sonra sırada sıralama işlemi var
items.sort()

#gösterim işlemi
print(' - '.join(items))
```

✓ 0.2s

```
durak - kitap - merhaba - okul
```

ÇÖZÜM 1.2

```
    items=[x for x in content.split(',')]
    items.sort()
    print(' - '.join(items))
```

✓ 0.1s

```
durak - kitap - merhaba - okul
```

SORU 2

Bir cümle kabul eden ve harf ve rakam sayısını hesaplayan bir program yazınız. Programa aşağıdaki girdinin sağlandığını varsayalım:

Selam Dünya! 123

Ardından, çıktı şöyle olmalıdır:

HARF: 10
RAKAM: 3

ÇÖZÜM 2

```
s = "Selam Dünya! 123"
d={"DIGITS":0, "LETTERS":0}
for c in s:
    if c.isdigit():
        d["DIGITS"]+=1
    elif c.isalpha():
        d["LETTERS"]+=1
    else:
        pass
print("LETTERS", d["LETTERS"])
print("DIGITS", d["DIGITS"])
```

✓ 0.1s

```
LETTERS 10
DIGITS 3
```

SORU 3

Bir robot, orijinal noktadan (0,0) başlayarak bir düzlemde hareket eder. Robot, verilen adımlarla YUKARI, AŞAĞI, SOL ve SAĞ yönünde hareket edebilir. Robot hareketinin izi aşağıdaki gibi gösterilir:
YUKARI 5
AŞAĞI 3
SOL 3
SAĞ 2

Yönden sonraki sayılar basamaklardır. Lütfen bir dizi hareket ve orijinal noktadan sonra mevcut konumdan olan mesafeyi hesaplayan bir program yazınız. Mesafe virgüllü ise, en yakın tamsayıyı yazdırmanız yeterlidir.

Örnek olarak aşağıdaki demetler programa girdi olarak verilirse:
YUKARI 5
AŞAĞI 3
SOL 3
SAĞ 2
Ardından, programın çıktısı şöyle olmalıdır: 2

ÇÖZÜM 3

```python
import math

pos = [0,0]
while True:
    s = input('Yön giriniz:')
    if not s:
        break
    movement = s.split(" ")
    direction = movement[0]
    if direction=="son":
        break
    steps = int(movement[1])
    if direction=="yukari":
        pos[0]+=steps
    elif direction=="asagi":
        pos[0]-=steps
    elif direction=="sol":
        pos[1]-=steps
    elif direction=="sag":
        pos[1]+=steps
    else:
        pass

print(int(round(math.sqrt(pos[1]**2+pos[0]**2))))
```

✓ 15.4s

Orta Şekerli Sorular

SORU 4

Find the closest palindrome number for your code:

Example
Input: 216
Output: 212

Input: 100
Output: 99

Input: 77
Output: 77

- Palindrome number is same from left-right side.
- If the number itself is a palindrome, return that number.
- If two palindrome numbers distance is same, choose smaller one.

ÇÖZÜM 4.1

```python
import sys

def areYouPalindrome(inputLen, myInput):
    returnVal = True
    if inputLen > 1:
        for i in range(inputLen//2 + 1):
            if myInput[i] != myInput[inputLen-i-1]:
                returnVal = False
    else:
        returnVal = False
    return returnVal

myInput = input("Bir deger giriniz:")
inputLen = len(myInput)

isPalindrome = True
counterForPalindrome = 0
ourPalindromeValue = 0

palindromeFound = False
while palindromeFound == False:
    if areYouPalindrome(inputLen, str(int(myInput) + counterForPalindrome)) == True:
        palindromeFound = True
        ourPalindromeValue = str(int(myInput) + counterForPalindrome)
        break
    elif areYouPalindrome(inputLen, str(int(myInput) - counterForPalindrome)) == True:
        palindromeFound = True
        ourPalindromeValue = str(int(myInput) - counterForPalindrome)
        break
    counterForPalindrome += 1

print(f"En yakın palindrom değer: {ourPalindromeValue}")
```

ÇÖZÜM 4.2

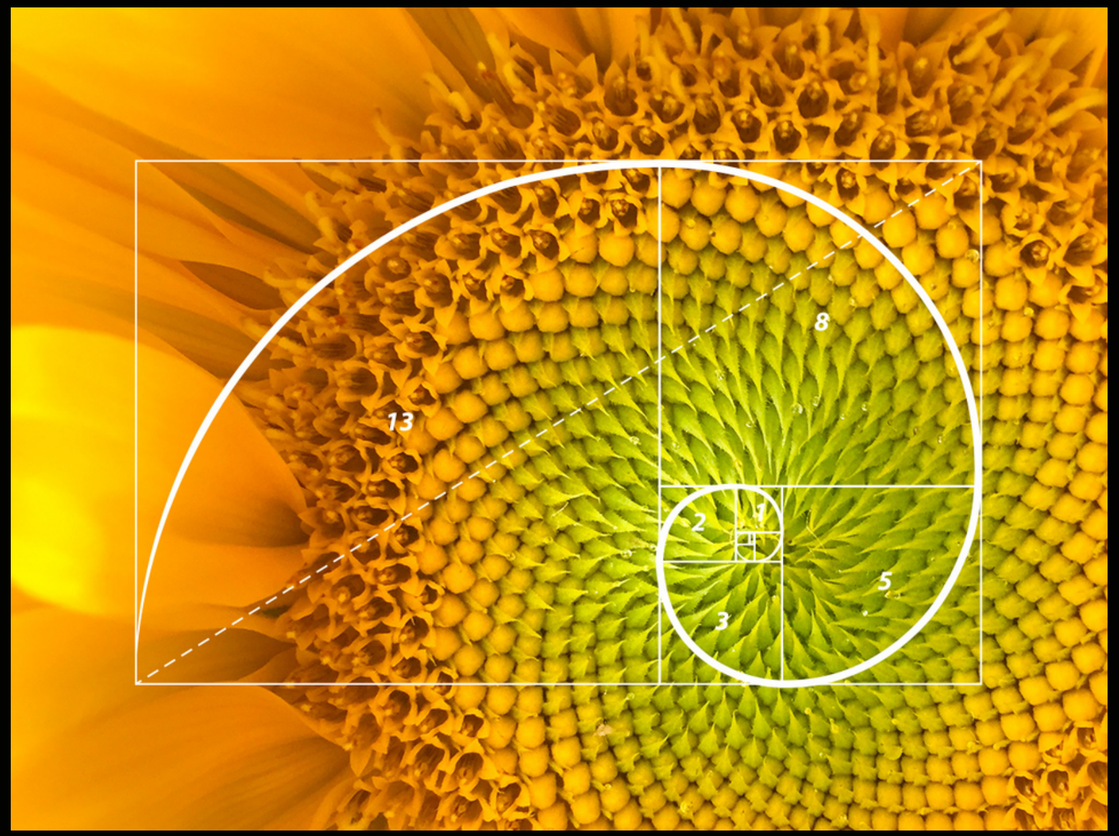
```python
def palin(n):
    u,d = n,n
    while(1):
        if u==int(str(u)[::-1]):
            return u
        elif d==int(str(d)[::-1]):
            return d
        u+=1
        d-=1
n=int(input())
print(palin(n))
```

SORU 5

Kullanıcıdan istenen değer kadar fibonacci dizisini gösteren bir program yazınız.

Input: 10
Output: 0 1 1 2 3 5 8 13 21 34

Input: 5
Output: 0 1 1 2 3



ÇÖZÜM 5

```python
nterms = int(input("Kaç değer istiyorsun? "))

# ilk iki değer
n1, n2 = 0, 1
count = 0

# girilen değer valid mi diye bakarız.
if nterms <= 0:
   print("Lütfen valid bir değer giriniz:")

elif nterms == 1:
   print("Fibonacci sekans1 ",nterms," tanedir:")
   print(n1)

else:
   print("Fibonacci sequence:")
   while count < nterms:
      print(n1)
      nth = n1 + n2
      # değerler güncellenir.
      n1 = n2
      n2 = nth
      count += 1
```

ÇÖZÜM 5.2

```python
def Fibonacci(n):

    if n < 0:
        print("Incorrect input")

    elif n == 0:
        return 0

    elif n == 1 or n == 2:
        return 1

    else:
        return Fibonacci(n-1) + Fibonacci(n-2)


deger = input("Değer giriniz:")
for i in range(int(deger)):
    print(Fibonacci(i))
```

SORU 6

Aç kapa parantezlerin kodlamada önemli olduğunu hepimiz biliyoruz.
Bu parantezlerin açıldıktan sonra kapanıp kapanmadığını kontrol eden kodu yazınız.

Örnekler:
Input: {{[[(())]]}}
Output: Balanced
Input: ()()(){[]
Output: Unbalanced
Input: [][]()(){[]()}
Output: Balanced
Input: {[(])}
Outout: Unbalanced

ÇÖZÜM 6

```python
open_list = ["[","{","("]
close_list = ["]","}",")"]

# Function to check parentheses
def check(myStr):
    stack = []
    for i in myStr:
        if i in open_list:
            stack.append(i)
        elif i in close_list:
            pos = close_list.index(i)
            if ((len(stack) > 0) and
                (open_list[pos] == stack[len(stack)-1])):
                stack.pop()
            else:
                return "Unbalanced"
    if len(stack) == 0:
        return "Balanced"
    else:
        return "Unbalanced"

```

```python
# Driver code
string = "{{[[(())]]}}"
print(string,"-", check(string))

string = "()()(()){[]"
print(string,"-", check(string))

string = "[][]()(()){[]()}"
print(string,"-",check(string))

string =  "{[(])}"
print(string,"-",check(string))
```

test

Zor Sorular

SORU 7

A valid postal code P have to fullfil both below requirements:

1. P must be a number in the range from 100000 to 999999 inclusive.
2. P must not contain more than one alternating repetitive digit pair.

Alternating repetitive digits are digits which repeat immediately after the next digit. In other words, an alternating repetitive digit pair is formed by two equal digits that have just a single digit between them.

For example:

```
121426 # Here, 1 is an alternating repetitive digit.
523563 # Here, NO digit is an alternating repetitive digit.
552523 # Here, both 2 and 5 are alternating repetitive digits.
```

Your task is to provide two regular expressions `regex_integer_in_range` and `regex_alternating_repetitive_digit_pair`. Where:

`regex_integer_in_range` should match only integers range from 100000 to 999999 inclusive

`regex_alternating_repetitive_digit_pair` should find alternating repetitive digits pairs in a given string.

Both these regular expressions will be used by the provided code template to check if the input string P is a valid postal code using the following expression:

```
(bool(re.match(regex_integer_in_range, P))
and len(re.findall(regex_alternating_repetitive_digit_pair, P)) < 2)
```

SORU 7

Input Format

Locked stub code in the editor reads a single string denoting P from stdin and uses provided expression and your regular expressions to validate if P is a valid postal code.

Output Format

You are not responsible for printing anything to stdout. Locked stub code in the editor does that.

Sample Input 0

```
110000
```

Sample Output 0

```
False
```

Explanation 0

1 1 `0`**`0`**`0`**`0`** : (`0`, `0`) and (**0**, **0**) are two alternating digit pairs. Hence, it is an invalid postal code.

Note:

A score of 0 will be awarded for using '`if`' conditions in your code.

You have to pass all the testcases to get a positive score.

ÇÖZÜM 7

```python
def inn(p):
    return 100000 <= int(p) <= 999999


def imm(p):
    l = []
    for i in range(len(p) - 2):
        comp = p[i] == p[i + 2]
        if (comp):
            l.append(p[i])
    return l


def is_valid(p):
    return (p.isdigit() and inn(p))

p = input()
if is_valid(p):
    returnVal = imm(p)
    if len(returnVal) < 1:
        print("NO digit is an alternating repetitive digit.")
    else:
        print(returnVal)
```

SORU 8

Mr. Anant Asankhya is the manager at the INFINITE hotel. The hotel has an infinite amount of rooms.

One fine day, a finite number of tourists come to stay at the hotel.

The tourists consist of:

→ A Captain.

→ An unknown group of families consisting of K members per group where $K \neq 1$.

The Captain was given a separate room, and the rest were given one room per group.

Mr. Anant has an unordered list of randomly arranged room entries. The list consists of the room numbers for all of the tourists. The room numbers will appear K times per group except for the Captain's room.

Mr. Anant needs you to help him find the Captain's room number.

The total number of tourists or the total number of groups of families is not known to you.

You only know the value of K and the room number list.

Input Format

The first line consists of an integer, K, the size of each group.

The second line contains the unordered elements of the room number list.

Constraints

$1 < K < 1000$

Output Format

Output the Captain's room number.

SORU 8

Sample Input

```
5
1 2 3 6 5 4 4 2 5 3 6 1 6 5 3 2 4 1 2 5 1 4 3 6 8 4 3 1 5 6 2
```

Sample Output

```
8
```

Explanation

The list of room numbers contains 31 elements. Since K is 5, there must be 6 groups of families. In the given list, all of the numbers repeat 5 times except for room number 8.

Hence, 8 is the Captain's room number.

SORU 8

```
# Sample Input
# 4
# 1 3 2 4 6 4 1 3 2 4 2 1 3 3 2 4 1
# Output: 6
```

ÇÖZÜM 8

```
5  k = int(input())
6  rooms = list(map(int, input().split()))
7  roomSet = set(rooms)
8  roomSum = sum(rooms)
9  roomSetSum = sum(roomSet) * k
10 captainsRoom = (roomSetSum - roomSum) // (k - 1)
11 print(captainsRoom)
```