Nested Loop dan String

Tutorial 4 [G, F] - Dasar-Dasar Pemrograman 1 Gasal 2019/2020

Selamat datang di Tutorial 4 DDP1! Sesi tutorial ini akan memperkenalkan kalian pada konsep *nested loop* dan *string*. Konsep yang sudah diajarkan sebelumnya, yaitu variabel, tipe data, branching tentunya tidak akan dilupakan pada tutorial kali ini, dan juga tutorial seterusnya.

Mohon kumpulkan semua file jawaban Anda dalam bentuk zip dengan format Lab4_Nama_NPM_KodeAsdos.zip. Contoh: Lab4_WindiChandra_1606862721_YE.zip

Judul	File Submission	Bobot Nilai
Latihan 11: Kapan Seriusnya?	latihan11.py	30
Latihan 12: Dasi Kupu-Kupu	latihan12.py	50
Latihan 13: Dasi Kupu-Kupu 2	latihan13.py	20
Challenge 3: PrimeGen	challenge3.py	0

Pengantar Nested Loop

Sama halnya seperti if, kita dapat melakukan loop di dalam loop.
Perhatikan contoh di bawah:

```
x = int(input("masukkan nilai x: "))
y = int(input("masukkan nilai y: "))
print()
for i in range (x):
    for j in range (y):
        print("(",i,",",j,")", sep="", end=" ")
    print()
```

Jika kita memasukkan nilai X=4, dan Y=5 program akan menghasilkan keluaran sebagai berikut:

```
tutorial@DDP1:~$ python tutorial.py
masukkan nilai x: 4
masukkan nilai y: 5

(0,0) (0,1) (0,2) (0,3) (0,4)
(1,0) (1,1) (1,2) (1,3) (1,4)
(2,0) (2,1) (2,2) (2,3) (2,4)
(3,0) (3,1) (3,2) (3,3) (3,4)
```

(diambil dari materi tambahan loop tutorial 3)

Pengantar Strings

Berikut adalah beberapa operasi yang dapat Anda lakukan pada tipe data string:

1. `x in z`

   Berguna untuk mengecek apakah string x merupakan bagian dari string z

```
x1 = 'a'
x2 = 'ak'
x3 = 'ak47'
z = 'akatsuki'

print(x1 in z)
print(x2 in z)
print(x3 in z)
```

```
True
True
False
```

2. `x.lower(), x.upper()`

   Berguna untuk menghasilkan versi string yang berisi lowercase semua atau uppercase semua.

```
x = 'Aku (ingin) dapat nilai 100 di Quiz 1'
y, z = x.lower(), x.upper()

print(x, y, z, sep='\n')
```

```
Aku (ingin) dapat nilai 100 di Quiz 1
aku (ingin) dapat nilai 100 di quiz 1
AKU (INGIN) DAPAT NILAI 100 DI QUIZ 1
```

3. Iterasi pada string

```
x = 'Pak Chanek mendapat bantuan dari Hackerman. '
x += 'Hackerman mencoba semua kemungkinan password hingga password yang benar
ditemukan.'

for c in x:
    print(c, end='-')

for kata in x.split():
    print(c)

for kalimat in x.split('.'):
    print(kalimat)
```

```
1
P-a-k- -C-h-a-n-e-k- -m-e-n-d-a-p-a-t- -b-a-n-t-u-a-n- -d-a-r-i- -H-a-c-k-e-r-m-a-n-.-
-H-a-c-k-e-r-m-a-n- -m-e-n-c-o-b-a- -s-e-m-u-a- -k-e-m-u-n-g-k-i-n-a-n- -p-a-s-s-w-o-r-d-
-h-i-n-g-g-a- -p-a-s-s-w-o-r-d- -y-a-n-g- -b-e-n-a-r- -d-i-t-e-m-u-k-a-n-.-2
Pak
Chanek
mendapat
bantuan
dari
Hackerman.
Hackerman
mencoba
semua
kemungkinan
password
hingga
password
yang
benar
ditemukan.
3
Pak Chanek mendapat bantuan dari Hackerman
 Hackerman mencoba semua kemungkinan password hingga password yang benar
ditemukan
```

Latihan #11: Kapan Seriusnya? (latihan11.py, skor: +30)

Konsep penting: nested loop, variable, branching, strings

Apakah Anda mengenal seseorang yang suka mengakhiri chat-nya dengan ‘wkwk’? Tahukah Anda seberapa sering ia tertawa dengan ‘wkwk’? Sekarang, Anda akan membuat program yang dapat menghitung berapa kali teman Anda tertawa dengan ‘wkwk’.

Program Anda akan menerima sebuah teks yang berisi satu atau lebih kata dalam huruf alfabet kecil. Setiap kata akan dipisahkan oleh tepat satu spasi. Sebuah kata dihitung sebagai tertawa ‘wkwk’ apabila kata tersebut mempunyai 2 atau lebih karakter dan dimulai dengan huruf ‘w’, kemudian dilanjutkan oleh ‘k’, kemudian ‘w’, kemudian ‘k’, dan seterusnya.

Berikut contoh kata yang termasuk tertawa ‘wkwk’ dan yang tidak:

- ‘wkwk’ → Termasuk
- ‘wkwkwkwkw’ → Termasuk
- ‘wk’ → Termasuk (walaupun tertawanya tidak terlihat niat, namun tetap tertawa)
- ‘wkw**kk**wk’ → Tidak termasuk (terdapat ‘k’ setelah huruf ‘k’)
- ‘**ao**wk**o**w**a**k**a**w**o**k’ → Tidak termasuk (ada karakter selain ‘w’ dan ‘k’)
- ‘kwk’ → Tidak termasuk (ingat harus dimulai dengan huruf ‘w’)
- ‘w’ → Tidak termasuk (kurang dari 2 huruf)

Selain dapat menghitung banyaknya tertawa ‘wkwk’, program Anda diharapkan bisa menampilkan kembali teksnya **dengan mengubah kata tertawa ‘wkwk’ menjadi huruf besar** agar pengguna program bisa memastikan bahwa program Anda benar.

Penilaian:

- 5 poin apabila program Anda dapat menghitung banyak kata ‘wkwk’ saja.
- 20 poin apabila Anda menampilkan banyak kata yang termasuk tertawa ‘wkwk’.
- 5 poin apabila Anda menampilkan teks kembali dengan mengubah kata tertawa ‘wkwk’ menjadi huruf besar.

Contoh interaksi program berada pada halaman selanjutnya.

Contoh interaksi program:

Masukkan teks:
yaampun ddp satu sumpah gampang banget wkwk gini doang semua orang juga bisa tapi kalo sarjana hahaha gak lucu banget lawakannya wkwkwkwkwkwkwkw apaan sih yaudah mending kita pulang naik kwk nanti turun di mana gitu sekalian main dulu wk terus juga huruf w tuh sebenernya huruf ke dua puluh tiga

Hasil:
yaampun ddp satu sumpah gampang banget **WKWK** gini doang semua orang juga bisa tapi kalo sarjana hahaha gak lucu banget lawakannya **WKWKWKWKWKWKWKW** apaan sih yaudah mending kita pulang naik kwk nanti turun di mana gitu sekalian main dulu **WK** terus juga huruf w tuh sebenernya huruf ke dua puluh tiga

Tertawa wkwk sebanyak 3 kali

Contoh interaksi program lain:

Masukkan teks:
bola bola apa yang mirip kucing jawabannya bola emon hehehe jayus banget wkwk

Hasil:
bola bola apa yang mirip kucing jawabannya bola emon hehehe jayus banget **WKWK**

Tertawa wkwk sebanyak 1 kali

Contoh interaksi program lain:

Masukkan teks:
apa yang baunya kayak cat merah tapi warnanya biru hayo jawabannya cat biru

Hasil:
apa yang baunya kayak cat merah tapi warnanya biru hayo jawabannya cat biru

Tertawa wkwk sebanyak 0 kali

Perhatikan bahwa tulisan tebal berwarna biru adalah masukan Anda, dan tulisan tebal berwarna hitam adalah kata yang termasuk tertawa ‘wkwk’.

Hint:
Perhatikan program berikut:

```
print('wk' * 4)
```

Output:
wkwkwkwk

Perhatikan lagi:

```
a = 'wkwkwk'

print('wk' * (len(a) // 2) == a)
```

Output:
True

Anda dapat memproses setiap kata dan cek apakah kata tersebut termasuk dalam tertawa ‘wkwk’ atau bukan.

Ohya, program berikut membaca sebuah teks, kemudian memproses setiap kata dan menampilkan kembali setiap kata dalam bentuk *uppercase* :)

```
teks = input('Masukkan teks:\n')
words = teks.split()


print('Hasil:')
for word in words:
    print(word.upper(), end=" ")
```

Kira-kira, apa saja yang harus ditambahkan pada program berikut?

Latihan #12: Dasi Kupu-Kupu (latihan12.py, skor: +50)

Konsep penting: pattern

Gambarlah sebuah gambar dasi kupu-kupu yang mempunyai lebar N! Berikut adalah contoh pola untuk berbagai macam N. **Lebar dasi adalah banyaknya karakter pada baris pertama dan baris terakhir, termasuk spasi.**

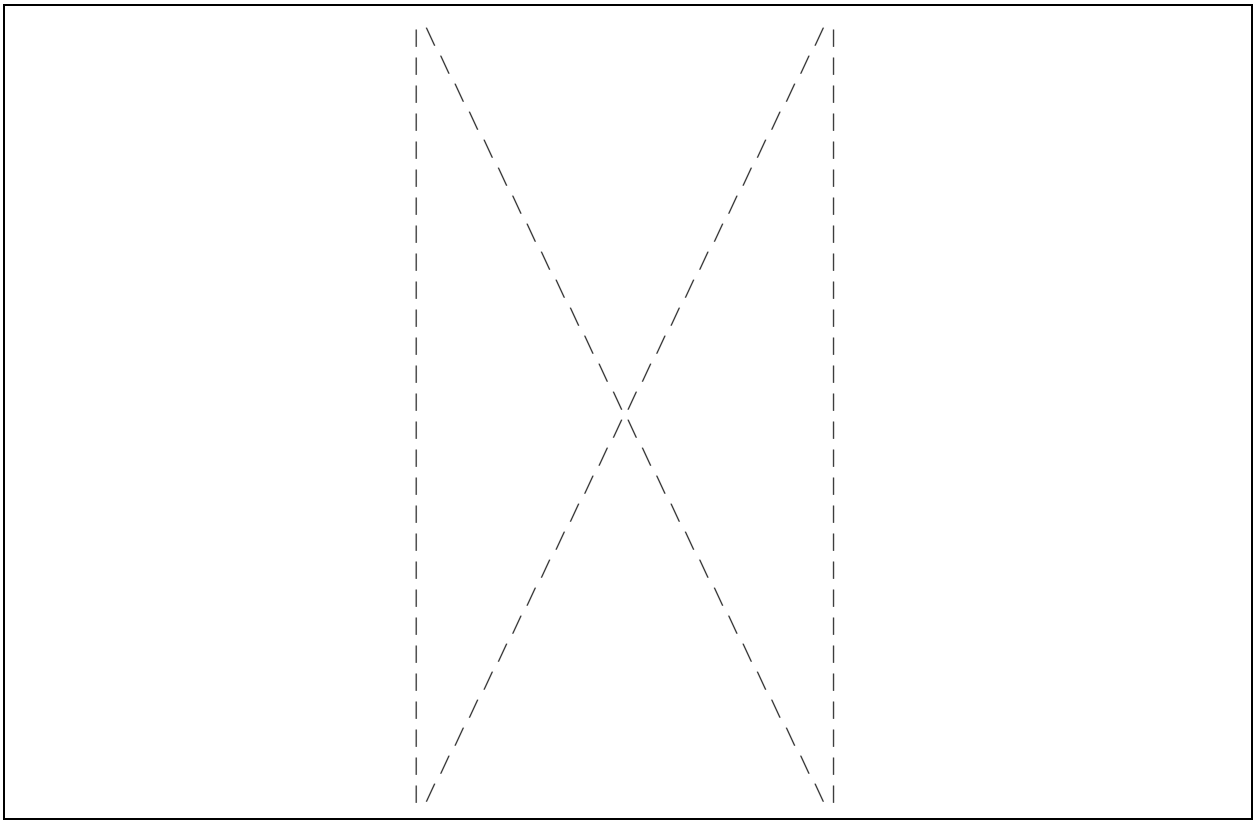
N = 4

```
|\/|
|/\|
```

N = 6

```
|\   /|
| \ / |
| /\  |
|/   \|
```

N = 30



Untuk mempermudah, Anda dapat mengasumsikan bahwa nilai N yang diberikan untuk program Anda merupakan kelipatan 2 dan bernilai 4 ke atas. Program Anda harus bisa menangani input setidaknya dari 4 sampai 50. Anda juga tidak perlu menangani masukan yang bukan kelipatan dua atau lebih kecil dari 4.

Contoh interaksi program:

```
Masukkan lebar dasi (harus kelipatan 2 dan >= 4):8
|\     /|
| \   / |
|  \/   |
|  /\   |
| /  \  |
|/    \|
```

Perhatikan bahwa tulisan tebal berwarna biru adalah masukan Anda.

Hint:
Cara mencetak simbol '\' → print("\\") (perhatikan \ dua kali)
Bagi 2 pola:

```
|\     /|
| \   / |
|  \/   |
```

Dan

```
|  /\   |
| /  \  |
|/    \|
```

Sebenarnya, bagian bawah merupakan bagian atas yang dibalik saja :)

Untuk mencetak pola bagian atas dengan lebar 8, maka kita akan membutuhkan tinggi sebanyak 3:

Cetak |, cetak \, cetak 4 spasi, cetak /, cetak |
Cetak |, cetak 1 spasi, cetak \, cetak 2 spasi, cetak /, cetak 1 spasi, cetak |
Cetak |, cetak 2 spasi, cetak \, , cetak /, cetak 2 spasi, cetak |

Untuk mencetak pola bagian atas dengan lebar 10, maka kita akan membutuhkan tinggi sebanyak 4:

Cetak |, cetak \, cetak 6 spasi, cetak /, cetak |
Cetak |, cetak 1 spasi, cetak \, cetak 4 spasi, cetak /, cetak 1 spasi, cetak |
Cetak |, cetak 2 spasi, cetak \, cetak 2 spasi, cetak /, cetak 2 spasi, cetak |
Cetak |, cetak 3 spasi, cetak \, , cetak /, cetak 3 spasi, cetak |

Perhatikan bahwa banyak spasi yang dicetak mempunyai pola tertentu.

Latihan #13: Dasi Kupu-Kupu 2 (latihan13.py, skor: +20)

Konsep penting: variable, nested loop, pattern

Modifikasi program anda pada latihan 12 untuk menampilkan angka berpola di dalam pola dasi kupu-kupu!

N = 4

```
  |\/|
  |/\|
```

N = 6

```
    |\  /|
    |1\/2|
    |3/\4|
    |/  \|
```

N = 30

```
                          |\                      /|
                          |1\                    /2|
                          |34\                  /56|
                          |789\                /012|
                          |3456\              /7890|
                          |12345\            /67890|
                          |123456\          /789012|
                          |3456789\        /0123456|
                          |78901234\      /56789012|
                          |345678901\    /234567890|
                          |1234567890\  /1234567890|
                          |12345678901\/23456789012|
                          |345678901234\/567890123456|
                          |7890123456789\/0123456789012|
                          |3456789012345/\6789012345678|
                          |901234567890/  \123456789012|
                          |34567890123/    \45678901234|
                          |5678901234/      \5678901234|
                          |567890123/        \456789012|
                          |34567890/          \12345678|
                          |9012345/            \6789012|
                          |345678/              \901234|
                          |56789/                \01234|
                          |5678/                  \9012|
                          |345/                    \678|
                          |90/                      \12|
                          |3/                        \4|
                          |/                          \|
```

Perlu diketahui sebelumnya, pola angka yang ditampilkan merupakan urutan 1, 2, 3 … 9, 0 ditulis dari kiri ke kanan, atas ke bawah dengan kembali ke 1 apabila sudah mencapai 0.

Untuk mempermudah, Anda dapat mengasumsikan bahwa nilai N yang diberikan untuk program Anda merupakan kelipatan 2 dan bernilai 4 ke atas. Program Anda harus bisa menangani input setidaknya dari 4 sampai 50. Anda juga tidak perlu menangani masukan yang bukan kelipatan dua atau lebih kecil dari 4.

Anda dapat melakukan *copy-paste* dari jawaban latihan 12 Anda.

Contoh interaksi program:

```
Masukkan lebar dasi (harus kelipatan 2 dan >= 4):8
|\    /|
|1\  /2|
|34\/56|
|78/\90|
|1/  \2|
|/    \|
```

Perhatikan bahwa tulisan tebal berwarna biru adalah masukan Anda.

[Bonus] Challenge #3: PrimeGen (challenge3.py, skor: +0)

Buatlah program yang dapat menampilkan seluruh bilangan prima di bawah 10.000.000 (10^7). Program Anda harus berjalan dalam waktu di bawah 10 detik.