Modul 2 – *Stack*
Praktikum Struktur Data - 2019

Buatlah suatu modul dengan nama *stack*, dengan fungsi-fungsi utama yang terdapat dalam modul adalah:

- stack( ) : inisialisasi stack kosong
- push(data) : push suatu data ke dalam stack
- pop ( ) : penghapusan data yang berada pada *top of the stack*. Return value berupa data yang dihapus dari stack tersebut
- peep ( ) : informasi yang terdapat pada top of the stack
- isEmpty( ) : memeriksa stack apakah dalam keadaan kosong
- size( ) : informasi jumlah data yang terdapat pada stack

Buatlah modul dengan nama *stackImplementation*, yang berisi fungsi-fungsi untuk implementasi-implementasi penggunaan konsep stack, antara lain :

1. reverseWord (kata) :
   Gunakan fungsi-fungsi yang terdapat pada modul stack untuk membalik suatu kata. Output dari fungsi reverseWord(kata) ini adalah kata yang sudah disusun secara terbalik, seperti berikut

```
In [5]: stack.reverseWord('kita')

Out[5]: 'atik'
```

2. decToBin(num) :
   Gunakan fungsi-fungsi yang terdapat pada modul stack untuk mengkonversi suatu bilangan decimal menjadi bilangan biner, seperti contoh berikut :

```
In [6]: stack.decToBin(12)
Out[6]: '1100'
```

3. evaluatePost(str):
   Gunakan fungsi-fungsi yang terdapat pada modul stack untuk mengevaluasi ekspressi matematika postfix, dengan return value berupa hasil operasi matematika, dengan ketentuan sebagai berikut :
   a. antara operand maupun operator, dipisahkan dengan spasi,
      misalkan : 8 17 + yang berarti 8 + 17
   b. Tambahkan pengecekan error, antara lain :
      i. Jika jumlah operand terlalu banyak
      ii. Jika jumlah operand terlalu sedikit
      iii. Jika terdapat operasi pembagian dengan nol, seperti 9 0 /, yang berarti 9:0

   Berikut adalah contoh penggunaan fungsi evaluatePost(str):

```
In [3]: a='6 0 81 *'
        evaluatePost(a)
Out[3]: 'Error : Terlalu banyak Operand'

In [4]: a='8 7 + 0 /'
        evaluatePost(a)
Out[4]: 'Error : Pembagian dengan nol'

In [5]: a='21 2 + * /'
        evaluatePost(a)
Out[5]: 'Error : Operand terlalu sedikit'

In [6]: a='10 11 1 + - 5 *'
        evaluatePost(a)
Out[6]: -10.0
```

4. parenthesesCheck(str):
   Gunakan fungsi-fungsi yang terdapat pada modul stack untuk pengecekan kurung pada ekspressi matematika, dengan return value berupa nilai True jika ekspressi matematika sudah benar, dan false jika masih terdapat kesalahan. Selain nilai True dan False, tambahkan pesan error jika terjadi kesalahan, antara lain :
   - Jumlah kurung buka terlalu banyak
   - Jumlah kurung tutup terlalu banyak
   - Kurung buka dan kurung tutup tidaklah cocok

   Contoh penggunaan fungsi dan output yang dihasilkan, dapat dilihat pada contoh berikut

```
In [3]: parenthesesCheck('{(3+5)*(9-2)}')
Out[3]: (True, 'Tidak ada Error')

In [4]: parenthesesCheck('(21/(56+2)')
Out[4]: (False, 'Jumlah Kurung Buka Lebih banyak')

In [5]: parenthesesCheck('(4-5)/12+7)')
Out[5]: (False, 'Jumlah Kurung Tutup lebih banyak')

In [6]: parenthesesCheck('67*(5+2)/(12-6}')
Out[6]: (False, 'Kurung Buka dan Kurang Tutup tidak Cocok')
```