Topik: Queue dan Stack

Tujuan Praktikum:
Mahasiswa memahami implementasi dan penggunaan ADT Stack dan Queue dengan representasi array dalam bahasa C.

PETUNJUK PRAKTIKUM:
1. Setiap ADT dibuat dengan format penamaan file sebagai berikut:
   a. Untuk file header : <namaADT>.h
   b. Untuk file realisasi : <namaADT>.c
   c. Untuk file driver/program utama : m<namaADT>.c

   Dengan:
   - <namaADT> : nama ADT dalam satu kata, contoh: point

   Contoh:
   point.h; point.c; mpoint.c
2. Untuk setiap file yang Anda buat, buat header sebagai berikut:

```
/* NIM/Nama  :                                                    */
/* Nama file :                                                    */
/* Topik     :                                                    */
/* Tanggal   :                                                    */
/* Deskripsi :                                                    */
```

3. Untuk setiap ADT, di-upload setelah dikompres menjadi 1 file dengan nama: p06<nim><namaADT>.zip.
   dengan:
   - <nim> : NIM Anda
   - <namaADT> : nama ADT dalam satu kata, contoh: point

   Contoh: p0613512500point.zip
4. Softcopy materi kuliah dan diktat, termasuk yang terkait dengan pemrograman dengan Bahasa C dapat dilihat pada situs **http://kuliah.itb.ac.id** pada link **IF2110/Algoritma dan Struktur Data**.
5. HANYA ADT YANG DAPAT DI-COMPILE YANG AKAN DIPERIKSA. File yang tidak dapat di-*compile* akan otomatis mendapatkan nilai 0.
6. Tugas ini bersifat INDIVIDUAL. Tidak ada toleransi bagi pencontek. Jika terbukti, baik yang dicontek maupun yang mencontek akan mendapatkan nilai 0.
7. Ikuti petunjuk asisten untuk pengumpulan tugas.

SELAMAT BEKERJA.

Soal 1. ADT Queue (Bobot : 50%)

a. Ambillah ADT Queue yang telah Anda selesaikan sebagai tugas pra-praktikum, dan salin lah menjadi ADT PrioQueue.
b. Queue dalam tugas ini akan digunakan untuk mensimulasikan antrian dengan prioritas (*priority queue*). Pada *priority queue*, setiap elemen *queue* memiliki prioritas, sehingga elemen dengan prioritas lebih tinggi diantrikan di depan elemen dengan prioritas lebih rendah. Setiap elemen *queue* terdiri dari dua komponen: ***prio*** yang menyatakan prioritas dengan nilai 1..3 (3 adalah prioritas tertinggi) dan ***nilai*** yang menyatakan nilai dari elemen.
c. Lakukan perubahan berikut terhadap ADT PrioQueue:
   - Ubahlah tipe elemen queue menjadi <prio: integer[1..3], nilai: integer>.
   - Lakukan modifikasi terhadap prosedur Add, sehingga proses penambahan elemen dilakukan sesuai dengan prioritas. Elemen baru akan diantrikan di depan seluruh elemen *queue* yang memiliki prioritas lebih rendah, dan di belakang seluruh elemen *queue* yang memiliki prioritas lebih tinggi atau sama.
     Contoh: PQ = [<3,10>, <2,4>, <2,7>, <1,15>]
     Add(PQ,<2,20>) → PQ = [<3,10>, <2,4>, <2,7>, <2,20>, <1,15>]
     Add(PQ,<1,30>) → PQ = [<3,10>, <2,4>, <2,7>, <1,15>, <1,30>]
   - Tambahkan prosedur PrintQueue yang berfungsi mencetak ke layar seluruh elemen *queue*, satu elemen per baris, prioritas dan nilai dipisah dengan spasi.
   - Jika *queue* kosong, pada layar hanya dicetak tanda #.

Format Input

- Baris pertama berisi sebuah bilangan N yang menandakan jumlah elemen yang queue (ukuran).
- Baris kedua sampai N+1 berisi sepasang bilangan <prio, nilai> yang menandakan prioritas serta nilai dari bilangan, masukkanlah seluruh bilangan tersebut pada Queue dengan prosedur Add yang baru.

Format Output

- Outputkanlah sejumlah N baris pasangan <prio, nilai>. Jangan lupa untuk memberikan newline pada setiap baris.
- Bila Queue kosong, outputkan '#'

Contoh:

Input	Output
4 1 15 2 4 2 7 3 10	3 10 2 4 2 7 1 15
4 1 15 2 7 2 4 3 10	3 10 2 7 2 4 1 15
0	#

Kumpulkanlah File (queue.h, queue.c serta mQueue.c) dalam 1 file *.zip (ikuti aturan penamaan halaman pertama)

Soal 2. ADT Stack (Bobot : 50%)

Dengan memanfaatkan ADT Stack yang telah Anda kerjakan sebagai tugas pra-praktikum dan **Mesin Token** yang disediakan asisten/ atau **Mesin Token** yang Anda buat minggu lalu, buatlah sebuah program yang membaca sebuah pita karakter (**stdin**) yang hanya berisi bilangan positif dan 0 serta operator '+', '-', '*', '/', dan '^' yang membentuk sebuah ekspresi aritmatika dalam notasi postfix dan mengevaluasi ekspresi yang terdapat pada pita karakter tersebut.
Secara lebih rinci, program memroses hasil dari mesin token sebagai berikut:

a. Apabila menerima token bilangan, maka bilangan yang dibaca ditampilkan di layar dan dimasukkan ke dalam stack.
b. Apabila menerima token operator, maka operan yang sesuai diambil dari stack, sub ekspresi yang harus dieksekusi dituliskan ke layar (format <operan1> <operator> <operan2>), hitung hasil sub ekspresi tersebut, tampilkan ke layar, dan masukkan ke dalam stack. Operan "/" diartikan sebagai pembagian bilangan bulat (div).
c. Apabila pembacaan pita karakter telah selesai, hasil akhir ekspresi dituliskan ke layar.

Program akan menuliskan pesan "EKSPRESI KOSONG" apabila pita karakter kosong.

Format Input

Sebuah baris berisi ekspresi aritmatika postfix. Di-*supply* melalui stdin. Input dijamin minimal terdiri dari sebuah bilangan atau kosong.

Format Output

Rincian evaluasi ekspresi

Contoh Input Output

Input	Output

`25 10 * 15 +.`	`25`
	`10`
	`25 * 10`
	`250`
	`15`
	`250 + 15`
	`265`
	`Hasil=265`

`4 5 * 3 10 * 1 3 + / +.`	`4`
	`5`
	`4 * 5`
	`20`
	`3`
	`10`
	`3 * 10`
	`30`
	`1`
	`3`
	`1 + 3`
	`4`
	`30 / 4`
	`7`
	`20 + 7`
	`27`
	`Hasil=27`

`25.`	`25` `Hasil=25`
`.`	`EKSPRESI KOSONG`

Kumpulkanlah File (**stack.h, stack.c, mesinkarakter.c, mesinkarakter.h mesintoken.h mesintoken.c serta mStack.c**) dalam 1 file *.zip (ikuti aturan penamaan halaman pertama)