Topik: ADT Array dan ADT Matriks

Tujuan Praktikum:
- Mengimplementasikan ADTArray dan ADT Matriks dengan representasi yang ditentukan 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 : 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: p03<nim><namaADT>.zip.
   dengan:
   - <nim> : NIM Anda
   - <namaADT> : nama ADT dalam satu kata, contoh: point

   Contoh: p0313509500point.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 ARRAY TERURUT (Bobot : 70%)
ADT Array, Representasi Implisit – Dinamik

Ambillah ADT array dengan representasi **implisit** dan alokasi memori **dinamik** yang telah Anda kerjakan sebagai tugas pra-praktikum. Anda ditugaskan untuk melakukan modifikasi ADT array tersebut menjadi **ADT array dengan elemen terurut membesar**. Modifikasi yang dilakukan adalah:

1. Pastikan Anda telah menyelesaikan semua fungsi/prosedur terkait dengan pemrosesan tabel dengan elemen terurut membesar sebagai berikut:

```
{ ********** TABEL DGN ELEMEN TERURUT MEMBESAR ********** }

function SearchUrut (T : TabInt, X : ElType) → IdxType
{ Prekondisi: Tabel boleh kosong. Jika tidak kosong, elemen terurut
membesar. }
{ mengirimkan indeks di mana harga X dengan indeks terkecil diketemukan }
{ mengirimkan IdxUndef jika tidak ada elemen tabel bernilai X }
{ Menghasilkan indeks tak terdefinisi (IdxUndef) jika tabel kosong }

function Max (T : TabInt) → ElType
{ Prekondisi : Tabel tidak kosong, elemen terurut membesar }
{ Mengirimkan nilai maksimum pada tabel}
function Min (T : TabInt) → ElType
{ Prekondisi : Tabel tidak kosong, elemen terurut membesar }
{ Mengirimkan nilai minimum pada tabel}
function MaxMin (T : TabInt) → <ElType, ElType>
{ Prekondisi : Tabel tidak kosong, elemen terurut membesar }
{ Mengirimkan nilai maksimum dan minimum pada tabel }

procedure Add1Urut (input/output T : TabInt, input X : ElType)
{ Menambahkan X tanpa mengganggu keterurutan nilai dalam tabel }
{ Nilai dalam tabel tidak harus unik. }
{ I.S. Tabel boleh kosong, boleh penuh. }
{      Jika tabel isi, elemennya terurut membesar. }
{ F.S. Jika tabel belum penuh, menambahkan X. }
{      Jika tabel penuh, maka tabel tetap. }
{ Proses : Search tempat yang tepat sambil geser }
{          Insert X pada tempat yang tepat tersebut tanpa mengganggu
keterurutan }

procedure Del1Urut (input/output T : TabInt, input X : ElType)
{ Menghapus X yang pertama kali (pada indeks terkecil) yang ditemukan }
{ I.S. Tabel tidak kosong }
{ F.S. Jika ada elemen tabel bernilai X , }
{      maka banyaknya elemen tabel berkurang satu. }
{      Jika tidak ada yang bernilai X, tabel tetap. }
{      Setelah penghapusan, elemen tabel tetap kontigu! }
{ Proses : Search indeks ke-i dengan elemen ke-i=X. }
{          Delete jika ada. }
```

2. Tambahkanlah fungsi/prosedur di bawah ini:

```
function SearchUrutB (T : TabInt, X : ElType) → boolean
{ Prekondisi: Tabel boleh kosong. Jika tidak kosong, elemen terurut
  membesar. }
{ Mengirimkan true jika X ada di dalam T dan false jika X tidak ditemukan
  di T. }
function SumTab (T : TabInt) → ElType
{ Prekondisi : Tabel T tidak kosong }
{ Mengirim hasil penjumlahan semua elemen dalam T }
```

```
procedure UpdateElmt (input/output T : TabInt, input i : IdxType, input X
: ElType)
{ I.S. : T terdefinisi, tidak kosong.
         i merupakan indeks valid dalam T.
         X terdefinisi. }
{ F.S. : T terurut membesar dengan X menjadi salah satu nilai elemen T }
{ Proses : Nilai elemen T pada indeks ke-i diubah menjadi X.
           Selanjutnya dilakukan pengurutan ulang terhadap T. }
procedure DelAllX (input/output T : TabInt, input X : ElType)
{ I.S. : T terdefinisi, boleh kosong. X terdefinisi. }
{ F.S. : Semua elemen T yang bernilai X dihapus dari tabel penampung, jika
         X ada di T. Jika X tidak ada, T tetap. }
```

Jika perlu membuat prosedur/fungsi tambahan, jangan lupa tuliskan definisi, spesifikasi, dan realisasi dari prosedur/fungsi tersebut.

3. Buatlah sebuah program utama yang memanfaatkan ADT array dengan elemen terurut membesar dengan urutan sebagai berikut:
   - Membuat sebuah tabel kosong.
   - Menentukan suatu nilai efektif tertentu (asumsikan masukan nilai efektif pasti > 0 dan <= IdxMax) dan kemudian mengisi tabel sebanyak nilai efektif tertentu. Dalam proses pengisian, pengguna boleh memasukkan nilai secara tidak terurut, tapi hasil akhirnya tabel tetap harus terurut. Dengan demikian, gunakan prosedur **Add1Urut**.
   - Menuliskan isi tabel ke layar.
   - Memasukkan pilihan menu 1 s.d. 4 (asumsikan masukan pilihan selalu benar) dengan aksi yang dilakukan adalah sbb:
     1. Menjumlahkan isi seluruh tabel dan menuliskan hasilnya ke layar.
     2. Membaca sebuah nilai X bertype ElType dan nilai indeks i bertype IdxType dan kemudian meng-update nilai elemen tabel pada indeks ke-i dengan nilai X serta menuliskan kembali isi tabel ke layar. Ingat, tabel selalu jadi terurut membesar. Jadi, gunakan prosedur **UpdateElmt**. Asumsikan bahwa masukan indeks selalu benar.
     3. Membaca sebuah nilai Y bertype ElType dan menghapus nilai Y pertama yang ditemukan dalam tabel serta menuliskan kembali isi tabel ke layar. Gunakan prosedur **Del1Urut**.
     4. Membaca sebuah nilai Z bertype ElType dan menghapus semua nilai Z yang ditemukan dalam tabel serta menuliskan kembali isi tabel ke layar. Gunakan prosedur **DelAllX**.

Kumpulkan ADT Array Terurut Anda dan drivernya (arrayterurut.h, arrayterurut.c, dan marrayterurut.c dengan penamaan sesuai standar praktikum)

Format masukan dan keluaran program sekaligus contoh (bagian yang digarisbawahi adalah masukan dari pengguna).
Perhatikan bahwa agar Anda bisa berhasil memanfaatkan autograder dengan baik, format harus benar-benar sama seperti di bawah (termasuk semua jumlah spasi, enter, tanda baca seperti sama dengan, koma, titik dua, dsb).

Contoh-1:

```
Nilai efektif tabel = 6
Elemen ke-1 = 10
Elemen ke-2 = 4
Elemen ke-3 = -1
Elemen ke-4 = 0
Elemen ke-5 = 2
Elemen ke-6 = 4
[1] -1
[2] 0
[3] 2
[4] 4
[5] 4
[6] 10
Pilihan menu (1-4) = 1
Hasil penjumlahan seluruh elemen tabel = 19
```

Contoh-2:

```
Nilai efektif tabel = 6
Elemen ke-1 = 10
Elemen ke-2 = 4
Elemen ke-3 = -1
Elemen ke-4 = 0
Elemen ke-5 = 2
Elemen ke-6 = 4
[1] -1
[2] 0
[3] 2
[4] 4
[5] 4
[6] 10
Pilihan menu (1-4) = 2
Nilai baru = -3
Indeks untuk update (1-6) = 3
[1] -3
[2] -1
[3] 0
[4] 4
[5] 4
[6] 10
```

Contoh-3:

```
Nilai efektif tabel = 6
Elemen ke-1 = 10
Elemen ke-2 = 4
Elemen ke-3 = -1
Elemen ke-4 = 0
Elemen ke-5 = 2
Elemen ke-6 = 4
[1] -1
[2] 0
[3] 2
[4] 4
[5] 4
[6] 10
Pilihan menu (1-4) = 3
Nilai yang dihapus = 4
[1] -1
[2] 0
[3] 2
[4] 4
[5] 10
```

Contoh-4:

```
Nilai efektif tabel = 6
Elemen ke-1 = 10
Elemen ke-2 = 4
Elemen ke-3 = -1
Elemen ke-4 = 0
Elemen ke-5 = 2
Elemen ke-6 = 4
[1] -1
[2] 0
[3] 2
[4] 4
[5] 4
[6] 10
Pilihan menu (1-4) = 4
Nilai yang dihapus seluruhnya = 4
[1] -1
[2] 0
[3] 2
[4] 10
```

Soal 2. ADT MATRIKS (Bobot : 30%)

Ambillah ADT Matriks yang sudah Anda kerjakan sebagai tugas pra-praktikum. Asisten akan menyediakan sebuah driver yaitu file **mmatriks.c** yang akan digunakan untuk memeriksa ADT Anda. Interaksi yang dilakukan oleh driver adalah sesuai dengan format sekaligus contoh di bawah. Bacalah source code tersebut baik-baik untuk melihat beberapa asumsi.

Periksalah apakah ADT Anda sudah berjalan dengan baik untuk driver tersebut dan periksalah apakah ADT Anda sudah lengkap dan sudah dapat memenuhi format di bawah. Perhatikan beberapa hal berikut:

- Ubahlah nama driver mmatriks.c dengan format sesuai standar praktikum dan jangan lupa periksa perintah *include file* di dalamnya.
- Perhatikan apakah semua ejaan penamaan type, fungsi dan prosedur sudah sesuai format dalam program driver.
- Untuk function perkalian 2 matriks, ubah namanya menjadi function **Kali2Matriks**. Ralat prekondisi:

  Tertulis:
  ```
  { Prekondisi : Ukuran baris efektif M1 = ukuran kolom efektif M2 }
  ```
  Seharusnya:
  ```
  { Prekondisi : Ukuran kolom efektif M1 = ukuran baris efektif M2 }
  ```

  Sesuaikan function **Kali2Matriks** Anda dengan hal ini.

- Prosedur **BacaMATRIKS**, yaitu dengan format dan contoh sbb. (yang diberi garis bawah adalah masukan pengguna):
  Misalnya ukuran baris = 3, dan ukuran kolom = 3

```
Baris ke-1
Kolom ke-1 = 0
Kolom ke-2 = 1
Kolom ke-3 = 2
Baris ke-2
Kolom ke-1 = 1
Kolom ke-2 = 1
Kolom ke-3 = 10
Baris ke-3
Kolom ke-1 = 0
Kolom ke-2 = 0
Kolom ke-3 = 0
```

- Prosedur **TulisMATRIKS**, yaitu dengan format dan contoh sbb:
  Misalnya ukuran baris = 3, dan ukuran kolom = 3

```
[0 1 2]
[1 1 10]
[0 0 0]
```

Kumpulkan ADT Matriks (matriks.h, matriks.c, dan mmatriks.c dengan penamaan sesuai dengan standar praktikum).

Format masukan dan keluaran program sekaligus contoh-contoh (bagian yang digarisbawahi adalah masukan dari pengguna):

Contoh-1:

```
Matriks 1
Ukuran efektif baris (1-100) = 3
Ukuran efektif kolom (1-100) = 2
Baris ke-1
Kolom ke-1 = 0
Kolom ke-2 = 0
Baris ke-2
Kolom ke-1 = 0
Kolom ke-2 = 0
Baris ke-3
Kolom ke-1 = 0
Kolom ke-2 = 0
[0 0]
[0 0]
[0 0]
Matriks 2
Ukuran efektif baris (1-100) = 3
Ukuran efektif kolom (1-100) = 2
Baris ke-1
Kolom ke-1 = -1
Kolom ke-2 = 1
Baris ke-2
Kolom ke-1 = 1
Kolom ke-2 = 2
Baris ke-3
Kolom ke-1 = 1
Kolom ke-2 = 4
[-1 1]
[1 2]
[1 4]
Pilihan menu (1-4) = 1
Hasil penjumlahan 2 matriks
[-1 1]
[1 2]
[1 4]
```

Contoh-2:

```
Matriks 1
Ukuran efektif baris (1-100) = 3
Ukuran efektif kolom (1-100) = 2
Baris ke-1
Kolom ke-1 = 0
Kolom ke-2 = 0
Baris ke-2
Kolom ke-1 = 0
Kolom ke-2 = 0
Baris ke-3
Kolom ke-1 = 0
Kolom ke-2 = 0
[0 0]
[0 0]
[0 0]
Matriks 2
Ukuran efektif baris (1-100) = 3
Ukuran efektif kolom (1-100) = 2
Baris ke-1
Kolom ke-1 = -1
Kolom ke-2 = 1
Baris ke-2
Kolom ke-1 = 1
Kolom ke-2 = 2
Baris ke-3
Kolom ke-1 = 1
Kolom ke-2 = 4
[-1 1]
```

```
[1 2]
[1 4]
Pilihan menu (1-4) = 2
Hasil perkalian 2 matriks
Ukuran efektif kolom matriks 1 tidak sama dengan ukuran efektif baris matriks
2, tidak bisa dikalikan
```

Contoh-3:

```
Matriks 1
Ukuran efektif baris (1-100) = 3
Ukuran efektif kolom (1-100) = 2
Baris ke-1
Kolom ke-1 = 0
Kolom ke-2 = 0
Baris ke-2
Kolom ke-1 = 0
Kolom ke-2 = 0
Baris ke-3
Kolom ke-1 = 0
Kolom ke-2 = 0
[0 0]
[0 0]
[0 0]
Matriks 2
Ukuran efektif baris (1-100) = 3
Ukuran efektif kolom (1-100) = 2
Baris ke-1
Kolom ke-1 = -1
Kolom ke-2 = 1
Baris ke-2
Kolom ke-1 = 1
Kolom ke-2 = 2
Baris ke-3
Kolom ke-1 = 1
Kolom ke-2 = 4
[-1 1]
[1 2]
[1 4]
Pilihan menu (1-4) = 3
Matriks 1 bukan matriks bujur sangkar
Matriks 1 bukan matriks simetris
Matriks 1 adalah matriks sparse
```

Contoh-4:

```
Matriks 1
Ukuran efektif baris (1-100) = 3
Ukuran efektif kolom (1-100) = 2
Baris ke-1
Kolom ke-1 = 0
Kolom ke-2 = 0
Baris ke-2
Kolom ke-1 = 0
Kolom ke-2 = 0
Baris ke-3
Kolom ke-1 = 0
Kolom ke-2 = 0
[0 0]
[0 0]
[0 0]
Matriks 2
Ukuran efektif baris (1-100) = 3
Ukuran efektif kolom (1-100) = 2
Baris ke-1
Kolom ke-1 = -1
Kolom ke-2 = 1
Baris ke-2
```

```
Kolom ke-1 = 1
Kolom ke-2 = 2
Baris ke-3
Kolom ke-1 = 1
Kolom ke-2 = 4
[-1 1]
[1 2]
[1 4]
Pilihan menu (1-4) = 4
Hasil transpose matriks 2
[-1 1 1]
[1 2 4]
```