SOAL ASESMEN 1 PRAKTIKUM STRUKTUR DATA

KELAS S1IF-12-04

Ketentuan dan Peraturan :

1. Hadir sebelum asesmen dimulai
2. Dilarang keras membagikan soal asesmen ke kelas lain (karena setiap kelas memiliki soal yang berbeda-beda)
3. Dilarang keras menggunakan AI (chatgpt, deepseek, gemini, blackbox, perplexity, claude, meta, dan sejenisnya) dalam mengerjakan soal asesmen
4. Jawaban dikumpulkan pada link google form yang sama seperti yang digunakan untuk mengumpulkan laprak
5. Waktu pengerjaan adalah 120 menit, bagi yang telat mengumpulkan akan diberikan pengurangan poin
6. Bagi yang ketahuan melanggar peraturan diatas akan dikenakan sanksi sesuai pedoman akademik yang berlaku
7. SELAMAT MENGERJAKAN 😊

Soal Nomor 1 (Single Linked List)

Bagian A :

Buatlah ADT Single Linked List (SLL) untuk menyimpan data produk toko dalam file "SLLInventory.h" sebagai berikut :

```
Type Struct Product <
    Nama : String
    SKU : String
    Jumlah : int
    HargaSatuan : Float
    DiskonPersen : Float (*nilai dalam persen (misal 10.5 untuk 10.5%))
    Type address : pointer/Node
>

Type Struct Node <
    info : Product
    next : address
>

Type List <
    head : address
```

Deklarasikan prosedur/fungsi berikut pada "SLLInventory.h" :

```
Function isEmpty( input List : List ) : boolean
Procedure createList( output L : List )
Function allocate( input P : Product ) : address
Procedure deallocate( input addr : address )
```

```
Procedure insertFirst( input L : List, in P : Product )
Procedure insertLast( input L : List, in P : Product )
Procedure insertAfter( input L : List, in Q : address, in P : Product ) //Q
adalah node setelah mana insert
Procedure deleteFirst( input L : List, out P : Product )
Procedure deleteLast( input L : List, out P : Product )
Procedure deleteAfter( input L : List, in Q : address, out P : Product )
Procedure updateAtPosition( input L : List, posisi : int )(data yang di-
update diinputkan secara dinamis / oleh user – ganti seluruh field Product
pada posisi tersebut)
Procedure viewList( input L : List ) (tampilkan Nama, SKU, Jumlah,
HargaSatuan, DiskonPersen, dan HargaAkhir pada setiap node; tampilkan juga
indeks posisi node dimulai dari 1)
Procedure searchByFinalPriceRange( input L : List, minPrice : Float,
maxPrice : Float ) (tampilkan semua produk yang memiliki HargaAkhir dalam
rentang inklusif [minPrice, maxPrice] dan tampilkan juga posisi tiap produk
yang ditemukan)
```

(Rumus HargaAkhir = HargaSatuan * (1 - DiskonPersen / 100))

Implementasikan semua fungsi dan prosedur tersebut pada file “SLLInventory.cpp”.

Kemudian lakukan hal-hal berikut di dalam file “main.cpp” :

1. Buat list kosong (createList).

2. Gunakan insertLast:
   Nama : Pulpen
   SKU : A001
   Jumlah : 20
   HargaSatuan : 2500
   DiskonPersen : 0

   Nama : Buku Tulis
   SKU : A002
   Jumlah : 15
   HargaSatuan : 5000
   DiskonPersen : 10

   Nama : Penghapus
   SKU : A003
   Jumlah : 30
   HargaSatuan : 1500
   DiskonPersen : 0

3. Tampilkan list

4. Lakukan deleteFirst 1x

5. Tampilkan list kembali.

6. Update data pada posisi ke-2
   Gunakan updateAtPosition() dengan data baru berikut:

   Nama : Stabilo
   SKU : A010
   Jumlah : 40
   HargaSatuan : 9000
   DiskonPersen : 5

7. Tampilkan list setelah update

8. Lakukan searching berdasarkan HargaAkhir dalam rentang
   min = 2000
   max = 7000

Bagian B:
Tambahkan prosedur pada ADT:
Procedure MaxHargaAkhir(L : List)
Fungsinya:

- Menampilkan data produk yang memiliki HargaAkhir terbesar.
- Menampilkan posisi produk tersebut.
- Bila ada lebih dari satu dengan harga akhir maksimum yang sama, tampilkan semua.
- Tambahkan pemanggilan MaxHargaAkhir(L) pada akhir main.cpp.

Soal Nomor 2 (Double Linked List)

Buatlah ADT Doubly Linked List (DLL) untuk menyimpan data lagu dalam file "DLLPlaylist.h" sebagai berikut:

```
Type Struct Song <
     Title : String
     Artist : String
     DurationSec : int
     PlayCount : int
     Rating : Float //skala 0.0 - 5.0
>

Type Struct Node <
     info : Song
     prev : address
     next : address
>

Type List <
     head : address
     tail : address
>
```

Deklarasikan fungsi/prosedur berikut pada “DLLPlaylist.h” :

```
Function isEmpty(L : List) : boolean
Procedure createList(L : out List)
Function allocate(S : Song) : address
Procedure deallocate(P : address)
Procedure insertFirst(L : input List, S : Song)
Procedure insertLast(L : input List, S : Song)
Procedure insertAfter(L : input List, Q : address, S : Song)
Procedure insertBefore(L : input List, Q : address, S : Song)
Procedure deleteFirst(L : input List, out S : Song)
Procedure deleteLast(L : input List, out S : Song)
Procedure deleteAfter(L : input List, Q : address, out S : Song)
Procedure deleteBefore(L : input List, Q : address, out S : Song)
Procedure updateAtPosition(L : input List, posisi : int)
Procedure updateBefore(L : input List, Q : address)
Procedure viewList(L : List)(tampilkan posisi, Title, Artist, DurationSec,
PlayCount, Rating, dan PopularityScore)
Procedure searchByPopularityRange(L : List, minScore : Float, maxScore :
Float)(tampilkan lagu-lagu yang memiliki PopularityScore dalam rentang
tersebut beserta posisinya)
```

Rumus Turunan (Wajib digunakan untuk output & searching)
PopularityScore = 0.8 * PlayCount + 20.0 * Rating

Implementasikan semua fungsi dan prosedur tersebut pada “DLLPlaylist.cpp”.

Instruksi “main.cpp”, lakukan hal berikut secara urut:
1. Buat list kosong (createList)

2. Masukkan 3 lagu menggunakan insertLast
   Title : “Senja di Kota”
   Artist : “Nona Band”
   DurationSec : 210
   PlayCount : 150
   Rating : 4.2

   Title : “Langkahmu”
   Artist : “Delta”
   DurationSec : 185
   PlayCount : 320
   Rating : 4.8

   Title : “Hujan Minggu”
   Artist : “Arka”
   DurationSec : 240
   PlayCount : 90
   Rating : 3.9

3. Tampilkan list

4. Lakukan deleteLast sebanyak 1x

5. Update node pada posisi ke-2
   Gunakan updateAtPosition() dengan data baru berikut:

   Title : “Pelita”
   Artist : “Luna”
   DurationSec : 200
   PlayCount : 260
   Rating : 4.5

6. Tampilkan list setelah update.

7. Operasi BEFORE
   Gunakan node pada posisi ke-2 untuk operasi before:
   a. insertBefore pada node posisi 2
      Title : “Senandung”
      Artist : “Mira”
      DurationSec : 175
      PlayCount : 120
      Rating : 4.0
      Kemudian tampilkan list setelah insertBefore.

   b. updateBefore pada node posisi 2
      Masukkan data baru (bebas), lalu tampilkan hasil update.

   c. deleteBefore pada node posisi 3
      Hapus node sebelum posisi ke-3, lalu tampilkan hasil.

8. Searching berdasarkan PopularityScore
   Gunakan rentang:
   min = 150.0
   max = 300.0

Soal Nomor 3 (Stack)

Bagian A :
Buatlah ADT Stack menggunakan ARRAY sebagai berikut di dalam file “StackMahasiswa.h” sebagai berikut :

```
Type Struct Mahasiswa <
        Nama : String
        NIM : String
        NilaiTugas : Float
        NilaiUTS : Float
        NilaiUAS: Float
```

```
>

const int MAX = 6

Type StackMahasiswa <
        dataMahasiswa[MAX] : Mahasiswa
        Top : int
>

Function isEmpty( input/output StackMHS : StackMahasiswa ) : boolean
Function isFull ( input/output StackMHS : StackMahasiswa ) : boolean
Procedure createStack ( input/output StackMHS : StackMahasiswa ) //terbentuk
stack dengan top = -1
Procedure Push ( input/output StackMHS : StackMahasiswa ) (data yang di-push
dilakukan secara dinamis atau diinputkan manual oleh user)
Procedure Pop ( input/output StackMHS : StackMahasiswa )
Procedure Update ( input/output StackMHS : StackMahasiswa, posisi : int )
(update berdasarkan posisi, data yang di-update diinputkan secara dinamis
atau diinputkan manual oleh user)
Procedure View ( input/output StackMHS : StackMahasiswa )
Procedure SearchNilaiAkhir ( input/output StackMHS : StackMahasiswa,
NilaiAkhirMin : Float, NilaiAkhirMax : Float ) //digunakan untuk searching
nilai akhir mahasiswa berdasarkan rentang tertentu, NilaiAkhir didapatkan
menggunakan rumus 20% NilaiTugas + 40% NilaiUTS + 40% NilaiUAS
```

Kemudian buatlah implementasi ADT Stack tersebut pada file “StackMahasiswa.cpp”.

Kemudian lakukan hal dibawah ini didalam file “main.cpp” :

1) buat stack kosong (createStack)
2) Input data mahasiswa dengan rincian sebagai berikut :
    1) Nama : Venti
       NIM : 11111
       NilaiTugas : 75,7
       NilaiUTS : 82,1
       NilaiUAS : 65,5
    2) Nama : Xiao
       NIM : 22222
       NilaiTugas : 87,4
       NilaiUTS : 88,9
       NilaiUAS : 81,9
    3) Nama : Kazuha
       NIM : 33333
       NilaiTugas : 92,3
       NilaiUTS : 88,8
       NilaiUAS : 82,4
    4) Nama : Wanderer

NIM : 44444
NilaiTugas : 95,5
NilaiUTS : 85,5
NilaiUAS : 90,5

5) Nama : Lynette
NIM : 55555
NilaiTugas : 77,7
NilaiUTS : 65,4
NilaiUAS : 79,9

6) Nama : Chasca
NIM : 66666
NilaiTugas : 99,9
NilaiUTS : 93,6
NilaiUAS : 87,3

3) Tampilkan stack yang sudah diinputkan data mahasiswa
4) Lakukan pop sebanyak 1x
5) Tampilkan stack yang sudah dilakukan pop 1x
6) Lakukan update data pada posisi ke 3 dengan rincian sebagai berikut :
Nama : Heizou
NIM : 77777
NilaiTugas : 99,9
NilaiUTS : 88,8
NilaiUAS : 77,7
7) Tampilkan stack yang sudah dilakukan update
8) Lakukan searching data mahasiswa yang memiliki NilaiAkhir dalam rentang 85,5 sampai 95,5; tampilkan pula posisi data ditemukan pada posisi keberapa (NilaiAkhir didapatkan menggunakan rumus 20% NilaiTugas + 40% NilaiUTS + 40% NilaiUAS)

Bagian B :
Tambahkan Prosedur MaxNilaiAkhir() yang digunakan untuk menampilkan data mahasiswa yang memiliki nilaiAkhir terbesar beserta posisinya berada di posisi keberapa. Kemudian tambahkan pemanggilan prosedur tersebut pada akhir main.cpp.

Soal Nomor 4 (Queue)

Sebuah perusahaan ekspedisi bernama Komaniya Express setiap harinya menerima banyak paket dari para pelanggan. Untuk memastikan paket diproses secara adil dan

teratur, perusahaan tersebut menerapkan struktur data antrian (Queue) dalam sistem pengiriman barang mereka.

Bagian A :
Buatlah ADT queue menggunakan ARRAY sebagai berikut di dalam file "QueuePengiriman.h" sebagai berikut :

```
Type Struct Paket <
        KodeResi : String
        NamaPengirim : String
        BeratBarang : Int //anggap satuan berat barang menggunakan kg
        Tujuan : String
>

const int MAX = 5

Type QueueEkspedisi <
        dataPaket[MAX] : Paket
        Head : Int
        Tail : Int
>

Function isEmpty( input/output Q : QueueEkspedisi ) : boolean
Function isFull ( input/ouput Q : QueueEkspedisi ) : boolean
Procedure createQueue ( input/ouput Q : QueueEkspedisi ) //terbentuk queue
dengan head = -1 dan tail = -1
Procedure enQueue( input/output Q : QueueEkspedisi ) (data yang dimasukkan
dilakukan secara dinamis atau diinputkan manual oleh user)
Procedure deQueue( input/output Q : QueueEkspedisi )
Procedure viewQueue( input/output Q : QueueEkspedisi )
```

(Note : queue yang dibuat menggunakan implementasi 1; yaitu head diam, tail bergerak)

Kemudian buatlah implementasi ADT Queue tersebut pada file "QueuePengiriman.cpp".

Kemudian lakukan hal dibawah ini didalam file "main.cpp" :

1) buat queue kosong (createQueue)
2) Buat tampilan menu seperti dibawah ini : (hint : gunakan percabangan switch-case)

```
--- Komaniya Ekspress ---
1. Input Data Paket
2. Hapus Data Paket
3. Tampilkan queue paket
Pilihan anda : ...
```

3) input data paket (enQueue) dengan rincian sebagai berikut :
    1) KodeResi :123456
       NamaPengirim : Hutao
       BeratBarang : 14
       Tujuan : Sumeru
    2) KodeResi : 234567
       NamaPengirim : Ayaka

BeratBarang : 10
Tujuan : Fontaine

3) KodeResi : 345678
NamaPengirim : Bennet
BeratBarang : 7
Tujuan : Natlan
4) KodeResi : 456789
NamaPengirim : Furina
BeratBarang : 16
Tujuan : Liyue
5) KodeResi : 567890
NamaPengirim : Nefer
BeratBarang : 6
Tujuan : Inazuma

4) Tampilkan queue yang sudah diinputkan data paket
5) Lakukan deQueue sebanyak 1x
6) Tampilkan queue yang sudah dilakukan deQueue 1x

Bagian B :

Tambahkan function TotalBiayaPengiriman() yang digunakan untuk menghitung total biaya pengiriman semua paket dalam queue (setiap 1 kg dikenakan biaya Rp. 8250). Kemudian pada main.cpp, tambahkan menu `4. Hitung Total Biaya Pengiriman` kemudian tampilkan total biaya pengiriman semua paket tersebut.