Question **4**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Merge Sorted List

Buatlah program `merge.c` yang mengimplementasikan `merge.h`

Diberikan juga file listlinier.h, listlinier.c, dan boolean.h

Deskripsi

`merge.c` berisi fungsi yang mengambil 2 buah list yang berurutan dan menyatukannya menjadi satu buah list yang berurutan.

Format Masukan

2 buah list linier yang berurutan dari kecil ke besar, L1 dan L2

Format Keluaran

1 list linier yang berurutan dari kecil ke besar

Batasan

ASUMSI :

1. List input adalah List linier yang berisi integer
2. List input mungkin kosong
3. List input mungkin memiliki lebih dari satu elemen yang sama, misalnya [1,1]
4. List input pasti berurutan dari kecil ke besar

CONTOH 1

input : L1 = [1,1,2,3] L2 = [1,2] output : [1,1,1,2,2,3]

CONTOH 2

input : L1 = [1,3] L2 = [2,4] output : [1,2,3,4]

C

merge.c

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	10	Accepted	0.00 sec, 1.66 MB
2	10	Accepted	0.00 sec, 1.52 MB
3	10	Accepted	0.00 sec, 1.65 MB
4	10	Accepted	0.00 sec, 1.55 MB
5	10	Accepted	0.00 sec, 1.60 MB

No	Score	Verdict	Description
6	10	Accepted	0.00 sec, 1.65 MB
7	10	Accepted	0.00 sec, 1.66 MB
8	10	Accepted	0.00 sec, 1.55 MB
9	10	Accepted	0.00 sec, 1.60 MB
10	10	Accepted	0.00 sec, 1.69 MB

Question **5**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

LRU Cache

Buat program dengan nama `cache.c`

Diberikan juga listlinier.h, listlinier.c, dan boolean.h

Deskripsi

Pada sistem operasi, akses memori pada RAM cukup lambat. Karena itu, digunakan cache yang jauh lebih cepat. Namun cache berukuran sangat kecil, jadi cache hanya menyimpan sedikit bagian dari RAM. Karena tidak semua nilai disimpan, saat dibutuhkan sebuah nilai x, cache bisa hit (cache memiliki nilai x) atau miss (cache tidak memiliki nilai x).

Salah satu implementasi cache adalah dengan skema LRU, yakni Least Recently Used. Pada skema ini, apabila nilai x tidak ada di cache, cache akan menghapus nilai di cache yang paling lama sudah tidak digunakan. Jadi, cache menyimpan daftar nilai, terurut dari yang paling baru digunakan, sampai yang paling lama digunakan. Cache dapat direpresentasikan sebagai sebuah list linier. Cache akan diinisialisasi dengan nilai 1 sampai N. Lalu, akan ada Q buah operasi pengambilan nilai x:

- Jika x ada di cache, nilai x dipindah ke depan cache.
- Jika x tidak ada di cache, nilai paling akhir dihapus dari cache dan x dimasukkan ke depan cache.
- Untuk setiap operasi, tuliskan apakah operasi "hit" atau "miss". Lalu, tuliskan isi list.

Format Masukan

Sebuah integer (N) sebagai panjang cache, sebuah integer (Q) sebagai jumlah operasi, kemudian setelahnya integer yang dicek pada cache.

Format Keluaran

String berisi "hit" atau "miss" dan isi cache.

Contoh Input/Output

Input

```
6
5
4
6
7
5
5
```

Output

```
hit [4,1,2,3,5,6]
hit [6,4,1,2,3,5]
miss [7,6,4,1,2,3]
miss [5,7,6,4,1,2]
hit [5,7,6,4,1,2]
```

Penjelasan

- Pada awalnya, cache berisi [1,2,3,4,5,6] karena N=6 sehingga diisi [1..N]
- Ada 5 operasi yang dilakukan karena Q = 5.
- Pada operasi pertama, 4 ada di cache.
- Pada operasi kedua, 6 juga ada di cache.
- Pada operasi ketiga, 7 tidak ada di cache, jadi 5 dihapus dan 7 ditambahkan ke depan cache.
- Pada operasi kelima, 5 sudah tidak ada di cache, jadi 3 dihapus dan 5 ditambahkan ke depan cache.
- Pada operasi keenam, 5 sudah ada di cache.

C

cache.c

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	7	Accepted	0.00 sec, 1.51 MB
2	7	Accepted	0.00 sec, 1.62 MB
3	7	Accepted	0.00 sec, 1.63 MB
4	7	Accepted	0.00 sec, 1.55 MB
5	7	Accepted	0.00 sec, 1.66 MB
6	7	Accepted	0.00 sec, 1.50 MB
7	7	Accepted	0.00 sec, 1.63 MB
8	7	Accepted	0.00 sec, 1.63 MB
9	7	Accepted	0.00 sec, 1.59 MB
10	7	Accepted	0.00 sec, 1.64 MB
11	7	Accepted	0.00 sec, 1.60 MB
12	7	Accepted	0.00 sec, 1.62 MB
13	16	Accepted	0.00 sec, 1.59 MB

Question **6**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Satomoto Nakashi

Deskripsi

Melihat cuitan kemarin siang di mana ada praktikannya yang berdoa agar tidak ada ***implementasi linked list*** di praktikum, Satomoto Nakashi tetiba terpikir untuk menemukan sebuah konsep yang, pada kemudian hari, dinamakan sebagai blockchain. Seperti namanya, blockchain merupakan sebuah konsep yang menggabungkan blok-blok data yang saling terhubung membentuk sebuah rantai yang tidak terputus.

Seperti takdir yang tidak dapat dihindari, keunikan dari blockchain justru terletak pada keterhubungan antar bloknya, di mana setiap blok memiliki keterkaitan dengan blok sebelumnya, membentuk suatu rantai yang saling terhubung dan tidak dapat dipisahkan. Implementasi dari konsep ini dapat direalisasikan dengan menggunakan struktur data ***linked list***, di mana setiap nodenya (yang diwakili dengan tipe BlockData) terhubung dengan node sebelumnya melalui pointer.

Umumnya, setiap blok baru yang ditambahkan ke dalam blockchain memiliki struktur sebagai berikut:

- Previous Hash (prevHash): Merupakan hash dari blok sebelumnya
- Data: Value yang diinput oleh user
- Timestamp: Waktu pembuatan blok yang diinput oleh user

Namun, khusus untuk block pertama (yang disebut juga sebagai Genesis block) akan memiliki struktur yang berbeda sebagai berikut:

- Tidak memiliki blok sebelumnya (prevHash = 0)
- Menyimpan value berupa jumlah ASCII dari setiap karakter dalam kata "Satomoto"
- Timestamp di-set ke 0 sebagai penanda block awal

Dengan menggunakan boolean.h, listlinierv2.c dan listlinierv2.h, buatlah program **blockchain.c** yang mengimplementasikan blockchain.h untuk mensimulasikan operasi-operasi dasar blockchain.

C

blockchain.c

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	5	Accepted	0.00 sec, 1.56 MB
2	5	Accepted	0.00 sec, 1.64 MB
3	5	Accepted	0.00 sec, 1.64 MB
4	5	Accepted	0.00 sec, 1.55 MB
5	5	Accepted	0.00 sec, 1.72 MB
6	5	Accepted	0.00 sec, 1.71 MB

No	Score	Verdict	Description
7	5	Accepted	0.00 sec, 1.63 MB
8	5	Accepted	0.00 sec, 1.64 MB
9	5	Accepted	0.00 sec, 1.65 MB
10	5	Accepted	0.00 sec, 1.66 MB
11	5	Accepted	0.00 sec, 1.61 MB
12	5	Accepted	0.00 sec, 1.66 MB
13	5	Accepted	0.00 sec, 1.64 MB
14	5	Accepted	0.00 sec, 1.67 MB
15	5	Accepted	0.00 sec, 1.66 MB
16	5	Accepted	0.00 sec, 1.51 MB
17	5	Accepted	0.00 sec, 1.62 MB
18	5	Accepted	0.00 sec, 1.63 MB
19	5	Accepted	0.00 sec, 1.67 MB
20	5	Accepted	0.00 sec, 1.54 MB

◄ Praktikum 9

Jump to...

Pra-Praktikum 10 ►