Started on	Monday, 7 November 2022, 9:14 AM
State	Finished
Completed on	Monday, 7 November 2022, 9:52 AM
Time taken	37 mins 46 secs
Grade	**300.00** out of 300.00 (**100**%)

Question **1**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

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.

Gunakan listlinier.h, listliner.c, dan kumpulkan **cache.c**

Contoh input / output:

Input	Output	Penjelasan
6 5 4 6 7 5 5	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]	N = 6 Q = 5 Pada awalnya, cache berisi [1,2,3,4,5,6]. Ada 5 operasi yang dilakukan. 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 keempat, 5 sudah tidak ada di cache, jadi 3 dihapus dan 5 ditambahkan ke depan cache. Pada operasi kelima, 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, 1.54 MB
2	7	Accepted	0.00 sec, 1.60 MB
3	7	Accepted	0.00 sec, 1.64 MB
4	7	Accepted	0.00 sec, 1.57 MB
5	7	Accepted	0.00 sec, 1.50 MB
6	7	Accepted	0.00 sec, 1.55 MB
7	7	Accepted	0.00 sec, 1.64 MB
8	7	Accepted	0.00 sec, 1.56 MB
9	7	Accepted	0.00 sec, 1.50 MB

No	Score	Verdict	Description
10	7	Accepted	0.00 sec, 1.67 MB
11	7	Accepted	0.00 sec, 1.64 MB
12	7	Accepted	0.00 sec, 1.62 MB
13	16	Accepted	0.00 sec, 1.71 MB

Question **2**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

List fibonacci adalah list yang elemennya merupakan jumlah dari 2 elemen sebelumnya. Pada awal program, program akan meminta jumlah elemen dari list fibonaci yang akan dibuat. Selanjutnya, program akan mengeluarkan output sesuai aturan berikut:

- Apabila jumlah elemen = 0, maka program akan langsung mengembalikan list kosong.
- Apabila jumlah elemen = 1, maka program akan meminta sebuah inputan yang menjadi elemen dalam list
- Apabila jumlah elemen >=2, maka program akan meminta dua buah inputan. Inputan pertama adalah elemen pertama dalam list dan inputan kedua adalah elemen kedua dalam list

Berikut adalah contoh input/output dari program

Input	Output
0	[]
1 2	[2]
1 4	[4]
2 0 1	[0,1]
4 3 4	[3,4,7,11]

Untuk mempermudah pengerjaan, silahkan lengkapi file template.c (boleh mengganti blok kode program yang telah ditulis pada template) dan submit dengan format nama listfibonacci.c

Note: Semua input berupa bilangan bulat positif

Diberikan ADT yang digunakan adalah:
- listlinier.h
- listlinier.c

C

template.c

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	6	Accepted	0.00 sec, 1.50 MB
2	6	Accepted	0.00 sec, 1.64 MB
3	6	Accepted	0.00 sec, 1.62 MB
4	6	Accepted	0.00 sec, 1.71 MB
5	6	Accepted	0.00 sec, 1.64 MB
6	6	Accepted	0.00 sec, 1.51 MB
7	6	Accepted	0.00 sec, 1.64 MB

No	Score	Verdict	Description
8	6	Accepted	0.00 sec, 1.71 MB
9	6	Accepted	0.00 sec, 1.64 MB
10	6	Accepted	0.00 sec, 1.64 MB
11	6	Accepted	0.00 sec, 1.50 MB
12	6	Accepted	0.00 sec, 1.64 MB
13	6	Accepted	0.00 sec, 1.71 MB
14	6	Accepted	0.00 sec, 1.63 MB
15	16	Accepted	0.01 sec, 1.64 MB

Question **3**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Buatlah implementasi **removeDuplicate.c** dari file header removeDuplicate.h!
Perhatian:
- List masukan sudah **terurut membesar**
- List hasil juga harus **terurut membesar**

- removeDuplicate.h
- listlinier.h
- listlinier.c
- boolean.h

C

removeDuplicate.c

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	20	Accepted	0.00 sec, 1.60 MB
2	20	Accepted	0.00 sec, 1.59 MB
3	20	Accepted	0.00 sec, 1.50 MB
4	20	Accepted	0.00 sec, 1.62 MB
5	20	Accepted	0.00 sec, 1.64 MB