Abstract Cache

Abstract classes in C++ can only be as base classes, so as a result they can have virtual functions that are not defined.

Cache is a component in which information is stored so that future requests for this information can be answered faster!

The information in the cache may be the result of previous calculations or a duplicate of the information stored elsewhere.

A cache hit has occurred when the requested data is available in the cache. And when it is not available, it is called a cache miss.

In the practice of cache hit, reading data is much faster than recalculating or reading information from disk or....

So, as a result, if we can respond to more requests by cache, we will give our system more speed.

There are different policies for queuing requests in the cache. One of these policies is "LEAST RECENTLY USED" or "LRU" in short, and it means that the newest request is always placed at the top of the queue and older requests are pushed back.

Questions

!All are well

Abstract Cache

Challenge: Encapsulation?
!?Are you kidding

All submissions

Final submissions

Scoreboard

For example, if we have a cache with the ability to store 5 keys and the initial state of our cache is as follows:

```
5 3 2 1 4
```

Now, if the next key that is requested is "1", the queue in our cache will be as follows:

```
1 5 3 2 4
```

And if the next key that is requested is "6", the queue in our cache will be as follows:

```
6 1 5 3 2
```

It can be seen that key 4 has been removed from the end of our queue because our cache size was 5 and it was no longer possible to hold it.

You are given an abstract *cache* class with the following data members and member functions:

- cp: cache memory volume
- set(): add new key to queue cache
- get() : The value available for that key in the cache if it exists. If that key is not available in the cache, it returns a negative value of one.

According to the said question, you need a data structure that equates each key to a data! It is up to you to choose or build this building.

Cache class code:

```
class Cache{

   protected:
   //data structure
   int cp;  //capacity
   virtual void set(int, int) = 0; //set function
   virtual int get(int) = 0; //get function

};
```

Input:

In the first line, you will be given the number N and M, which respectively represent the number of Set/Get commands and the amount of cache memory. In the next N line, first the name of the command and then the items needed for each command are entered.

At first, the cache is completely empty and there is no data in it.

Output:

The output of your code is actually the output of the Get command.

Example:

Input:

```
3 1
set 1 2
get 1
get 2
```

Output:

```
2
-1
```

POST AN ANSWER TO THIS QUESTION

.The training period is over

Employment platform

Quera Jr

Proudly made in Iran 1401 - 1394