**Emails, emails everywhere!**

https://hrcdn.net/s3_pub/hr-avatars/6df99f5d-1823-49c0-9679-7fae54add041/150x150.png**by [Han\_V](https://www.hackerrank.com/Han_V)**

* [**Problem**](https://www.hackerrank.com/contests/booking-womenintech/challenges/emails-emails-everywhere)
* [**Submissions**](https://www.hackerrank.com/contests/booking-womenintech/challenges/emails-emails-everywhere/submissions)
* [**Leaderboard**](https://www.hackerrank.com/contests/booking-womenintech/challenges/emails-emails-everywhere/leaderboard)
* [**Discussions**](https://www.hackerrank.com/contests/booking-womenintech/challenges/emails-emails-everywhere/forum)

Here at [Booking.com](https://www.hackerrank.com/external_redirect?to=http://www.booking.com/) we want to inspire our customers, and emails with destination recommendations can do the job. So, we want to send recommendation emails to our customers and we ask you to implement a temporary storage/buffer system so that we don’t overload our send servers.

This system is designed to have a very simple API that boils down to accepting 2 types of requests:

* **store {email\_content} {urgency}** - a request from email generator to temporarily store a rendered email
* **get\_next\_email** - a request from send system for the next email to send, expects **{email\_content}** as a response

Now given that we send millions of emails everyday, we can safely say we have a few thousands in our storage system at any point time. The question is which emails do we prefer to send out first? For that, we have an urgency rating for each email (as specified by the **store** request parameters). Send priorities are based on urgency rating and work as follows:

* If ui > uj => ith should be sent earlier than jth email
* If ui == ui => follow the [FIFO](https://www.hackerrank.com/external_redirect?to=https://en.wikipedia.org/wiki/FIFO_%28computing_and_electronics%29) or the order of emails coming in

Your code will get one request per line that will resemble API requests to answer. Obviously, we don't want to spam our users so once email is sent to the send system it should be removed from the storage.

**Input Format**

The first line contains N, number of store/fetch queries that will follow.

Each of the next N lines contains requests to your system of the following format (one per line):

* **store {email\_content} {urgency}**, where email\_content is a String consisting of [a-zA-Z0-9\_] and urgency is a positive integer
* **get\_next\_email**

**Constraints**

**Output Format**

For each **get\_next\_email** request encountered print out **{email\_content}** of the email that should be sent out next. If there are no outstanding emails print "-1" string on a newline.

**Sample Input 0**

3

store email1 1

store email2 10

get\_next\_email

**Sample Output 0**

email2

<https://www.hackerrank.com/contests/booking-womenintech/challenges/emails-emails-everywhere>

from Queue import PriorityQueue

class **MyPriorityQueue**(PriorityQueue):

def **\_\_init\_\_**(*self*):

PriorityQueue.\_\_init\_\_(*self*)

*self*.counter = 0

def **put**(*self*, item, priority):

PriorityQueue.put(*self*, (priority, *self*.counter, item))

*self*.counter += 1

def **get**(*self*, \*args, \*\*kwargs):

\_, \_, item = PriorityQueue.get(*self*, \*args, \*\*kwargs)

return item

queue = MyPriorityQueue()

*'''*

*s = [*

*"store segundo 2",*

*"store primero1 1",*

*"store primero2 1",*

*"store primero3 1",*

*"store primero4 1",*

*"store tercero 45",*

*"get\_next\_email",*

*"get\_next\_email",*

*"get\_next\_email" ,*

*"get\_next\_email" ,*

*"get\_next\_email" ,*

*"get\_next\_email" ,*

*"get\_next\_email" ,*

*"get\_next\_email" ,*

*"store cuarto 4",*

*"get\_next\_email",*

*"store cinco 5",*

*"get\_next\_email"]*

*'''*

n = int(raw\_input())

for i in range(n):

#print e

elem = raw\_input()

partes = elem.split(*' '*)

if partes[0] == *"store"*:

key = int(partes[2])\* (-1)

email = partes[1]

queue.put(email , key)

elif elem == *"get\_next\_email"*:

if not queue.empty():

em = queue.get()

print em

else:

print *"-1"*