General Note:

- Read and understand the question carefully.
- For implementation, you are permitted to use the code that you have submitted as an assignment.
- You are not permitted to use global variables and/or static variables, unless specified.
- Assume that all the inputs in the test cases are valid.

1. In a military area, group A wants to communicate a secret message to group B. Rather than directly sending the whole sentence, they send each of the consecutive words of the message on the consecutive days. Everyday, at different times an SMS that contains an alphabet and a positive number is sent to group B. The numbers sent on a day are distinct. The same alphabet may be sent multiple times, but along with different numbers.

   Group B noted down all the received alphabets and associated numbers. Due to network problems, they did not receive the messages on some days. On such days, they marked the alphabet and number as '$*$' and a 0, respectively. They can obtain the word sent on a day by assembling the alphabets received on that day in the increasing order of the associated numbers.

   Given the alphabet-number pairs in each message in the order of days, your job is to help group B in constructing the message by performing the following tasks:

   - Find all the words communicated to group B using priority queue implemented as a heap (array of structures).
   - Concatenate the words received on all days to form the entire message.

   Use the following structure to store an alphabet and its associated number.

```
struct SMS
{
  char alpha;
  int num;
};
```

   Use the following global variable to store the decoded message.

```
char message[10000];
```

   Your program should implement the following functions using priority queue as per the given function prototypes:

   - $main()$: Repeatedly read a character 'r', 'd', or 'p' from the console and call the corresponding functions, as described below, until character 't' is encountered. [2 Marks]
   - $insert\_queue(Q, sms, n)$: Insert sms into the priority queue Q with current length n by using the concept of insertion into a heap as mentioned in CLRS, without storing it into another array. After the insertion, print the alphabets in Q in the increasing order of their position in the array, separated by a space. [2 Marks]
   - $decode\_word(Q, n)$: If the queue is empty, print "*" and append "*" to $message$. Otherwise, decode the word of length n from the priority queue Q using the extract function of heap, print it, and append it to $message$. [2 Marks]

Input/Output Format

The input consists of multiple lines. Each line starts with a character from $\{r, d, p, t\}$ followed by zero or one character and integer.

- Character 'r' : Character 'r' will be followed by a character $c \in [a-z, A-Z, *]$ and a non-negative integer $i \in [0, 10^5]$ representing the alphabet and associated number, respectively. Read c and i as an sms (variable of `struct SMS`) and if they are not '$*$' and 0, respectively, call function $insert\_queue()$.
- Character 'd' : Character 'd' represents the end of a day. Call function $decode\_word()$.
- Character 'p' : Print $message$.
- Character 't' : Terminate the program.

Input 1

```
r a 253
r S 100
r f 333
r e 599
d
r a 999
r r 8679
r W 777
d
p
t
```

Output 1

```
a
S a
S a f
S a f e
Safe
a
a r
W r a
War
Safe War
```

Input 2

```
r y 989
r p 670
r o 900
r d 390
r e 599
r l 888
d
r * 0
d
r o 994
r o 978
r t 56
r r 787
r s 2100
r p 1080
d
p
t
```

Output 2

```
y
p y
p y o
d p o y
d e o y p
d e l y p o
deploy
*
o
o o
t o o
t r o o
t r o o s
t r o o s p
troops
deploy * troops
```