Problem: Happy Lady Bugs

Happy Ladybugs is a board game having the following properties:

- The board is represented by a string, , of length . The character of the string, , denotes the cell of the board.
    - If is an underscore (i.e., `_`), it means the cell of the board is empty.
    - If is an uppercase English alphabetic letter (i.e., `A` through `Z`), it means the cell contains a ladybug of color .
    - String will not contain any other characters.
- A ladybug is *happy* only when its left or right adjacent cell (i.e.,) is occupied by another ladybug having the same color.
- In a single move, you can move a ladybug from its current position to any empty cell.

Given the values of and for games of Happy Ladybugs, determine if it's possible to make all the ladybugs happy. For each game, print `YES` on a new line if all the ladybugs can be made happy through some number of moves; otherwise, print `NO` to indicate that no number of moves will result in all the ladybugs being happy.

Input Format

The first line contains an integer, , denoting the number of games. The subsequent lines describes a Happy Ladybugs game in the following format:

1. The first line contains an integer, , denoting the number of cells on the board.
2. The second line contains a string, , describing the cells of the board.

Constraints

-
-
- It is guaranteed that string consists of underscores and/or uppercase English alphabetic letters (i.e., `_` and `A`through `Z`).

Output Format

For each game, print `YES` on a new line if it is possible to make all the ladybugs *happy*; otherwise, print `NO`.

Sample Input 0

```
4
7
RBY_YBR
6
X_Y__X
2
__
6
B_RRBR
```

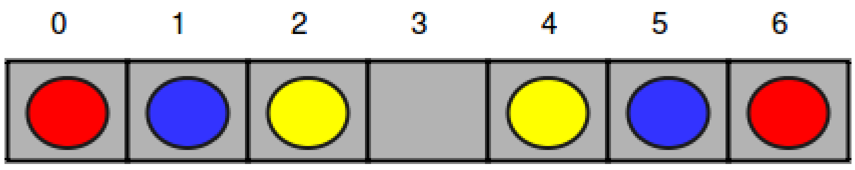
Sample Output 0

```
YES
NO
YES
YES
```

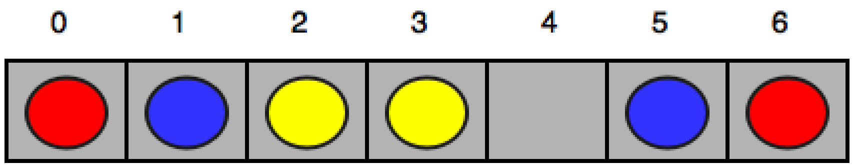
Explanation 0

The first three games of Happy Ladybugs are explained below:

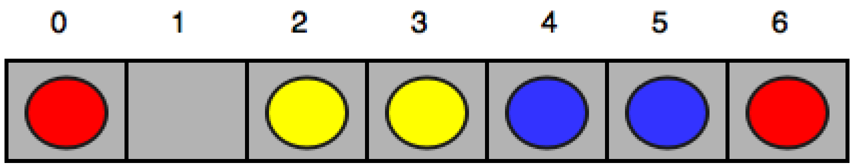
1. Initial board:

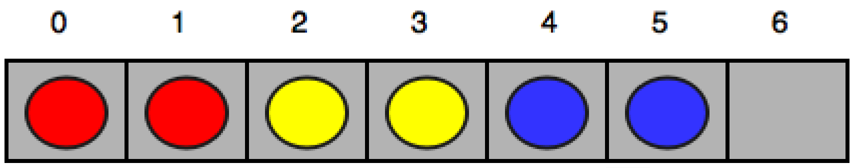
   After the first move:

   After the second move:

   After the third move:

   Now all the ladybugs are happy, so we print `YES` on a new line.
2. There is no way to make the ladybug having color `Y` happy, so we print `NO` on a new line.
3. There are no unhappy ladybugs, so we print `YES` on a new line.

Solution

```
string  sort(string  &str)     //Arranges same color bugs together
      {  if(str.find('_')<100)     //no empty slot, cannot Move
             {  int  length=str.length();
                for(int  i=0;  i<length-1;  i++)
                       {  for(int  j=0;  j<length-1-i;  j++)
                             {  if(str[j]>str[j+1])
                                         {
                                               char  temp=str[j];
                                               str[j]=str[j+1];
                                               str[j+1]=temp;
                                         }
                             }
                       }
            }
            return  str;
      }

//-----------------------------------------------------
int  happyCheck(string  str)     //Checks for happy Bugs
      {
            int  length=str.length();
            for(int  i=0;  i<length;  i++)
                   {  if(str[i]!='_')
                      {  int  count=0;
                         if(i==0  &&  str[i]==str[i+1])  {count+=1;  }
                         else  if(str[i]==str[i+1]  ||  str[i]==str[i-1]){count++;}
                   else  if(i==length-1  &&  str[i]==str[i-1]){count+=1;}

                         if(count!=1){return  0;}     //no happy bug found Abort
                      }
                   }
            return  1;
      }
//-----------------------------------------------
```

```
int  main()
     {
          int  testCases,  size;
          string  str;
          cin>>testCases;
          for(int  i=0;  i<testCases;  i++)
               {
                    cin>>size  >>str;
                    (happyCheck(sort(str))==1  ?  cout<<"YES"  :  cout<<"NO");
                    cout<<endl;
               }
          return  0;
     }
```

`’ Anshul AgGarwal