PROTECT THE PRIVACY

****
Author: Awadhesh Kumar
****
Explaination:

We can transform our word in binary notation, we can do it easily, because 64 = 26. Move through the bits of this number: if bit is equal to 0, then we can have 3 different optinos of this bit in our pair of words: 0&1, 1&0, 0&0, else we can have only one option: 1&1. So the result will be 3*nullbits*, where *nullbits* — is amount of zero bits.

Code:

```
#include <iostream>
#include <stdio.h>
#include <string>
#include<bits/stdc++.h>
#define MOD 1000000007
using namespace std;
typedef long long ll;
ll i,j,n,h,ans,x,cur_h,k;
string s;
string pattern;
ll symbol_val[305];
int main()
{
	cin>>s;
		for (char i = '0'; i <= '9'; i++)
			pattern.push_back(i);
		for (char i = 'A'; i <= 'Z'; i++)
			pattern.push_back(i);
		for (char i = 'a'; i <= 'z'; i++)
			pattern.push_back(i);
		pattern.push_back('-');
		pattern.push_back('_');
		for (i = 0; i < 64; i++)
			symbol_val[pattern[i]] = i;
		ll ans = 1;
		for (i = 0; i < s.size(); i++)
		{
			ll x = symbol_val[s[i]];
			//cout<<"x "<<x<<endl;
			for (j = 0; j < 6; j++)
				if ((x&(1<<j)) == 0)
				{
				  ans = (ans*3)%MOD;
				  //cout<<"ans "<<ans<<endl;
				}
		}
		cout << ans << endl;
	return 0;
}
```

Time Complexity: O(|s|), where |s| is the length of the string