```
     //
     //  main.cpp
     //  CSES Palindrome Reorder
     //
     //  Created by Harsh Anand on 22/10/2023.
     //

#include <bits/stdc++.h>
using namespace std;
using namespace chrono;
#define H_A code by Harsh Anand

#define Inf 2147483647
#define Pi acos(-1.0)

#define pb(x) push_back(x)

#define  sc(x) cin>>x;
#define pt(x) cout<<x<<"\n";
#define debug(x)  cout << #x << " = "<< x << "\n";
#define debug_2(x,y) cout << #x<<" = " << x <<" "<< #y<< " = " << y <<"\n";
#define no cout<<"NO\n"
#define yes cout<<"YES\n"
#define br cout<< "\n" //yey html


string btos( string se){
     string s= bitset<32> (se).to_string();
     return s;
}

uint64_t s_to_d( string st){
     uint64_t number;
     number = strtoull(st.c_str (),NULL,2);
     return number;
}


struct node {
     char data;
     struct node* next;
};

typedef  struct node *linked_list;

linked_list head, tail;

void push_front(char c){
     linked_list temp = (linked_list) malloc( sizeof(linked_list));
     temp->data=c;
     temp->next=nullptr;
     if (head==NULL) {
          head=tail=temp;
     }else{
          temp->next=head;
```

```
          head=temp;
     }
     return;
}//O(1)


void push_backk(char c,long long int x){
     while (x--) {
          linked_list temp = (linked_list) malloc( sizeof(linked_list));
          temp->data=c;
          temp->next=nullptr;
          if (!head) {
               head=tail=temp;
          }else{
               tail->next=temp;
               tail=temp;
          }//O(n)
     }
     return;
}



void print(linked_list temp){
     while (temp) {
          cout<<temp->data;
          temp=temp->next;
     }
}//O(n)

void insert_mid(char c, long long int index, long long int x){
     linked_list temp=head;
     for(long long int i=1; i<index-1; ++i){
          temp=temp->next;
     }
     while (x--) {
          linked_list n= (linked_list) malloc(sizeof(linked_list));
          n->data=c;
          n->next=temp->next;
          temp->next=n;;
     }
}//O(n)



void solve(string s){
     unordered_map<char, int> m;
     for (int i=0; s[i]; ++i) {
          m[s[i]]++;
     } // seprating the string //O(n)

     long long int z=0;
     long long int y=0;

     priority_queue<pair<long long int , long long int>> order_queu;
```

```
     for(auto it : m) {
          order_queu.push({it.second,it.first});
     } //O(n)


     for(auto x:m){
          if (x.second%2==0) {
               z++;
          }else
               y++;
     }//O(n)


     bool che=true;
     float prev=0, x=0;

     if(y==1 || y==0){
          while (!order_queu.empty()) {
               x+=order_queu.top().first;
               prev= ceil (x/2);
               if (che) {
                    push_backk(order_queu.top().second,
                     order_queu.top().first);
                    che=false;
               }else{
                    if (order_queu.top().first&1) {

                         insert_mid(order_queu.top().second, prev,
                          order_queu.top().first);
                    }else{
                         uint64_t x=(order_queu.top().first)/2;
                         while (x--) {
                              push_backk(order_queu.top().second, 1);
                              push_front(order_queu.top().second);
                         }//O(n)
                    }
               }
               order_queu.pop();
          }
          print(head);
     }//O(26)
     else{
          cout<<"NO SOLUTION\n";
     }



     return;
}




int main(){
```

```
#ifndef ONLINE_JUDGE
               freopen("/Users/harshanand/Desktop/C++ file/CSES Palindrome
                Reorder/CSES Palindrome Reorder/dick.in","r",stdin);
//               freopen("/Users/harshanand/Desktop/C++ file/CSES
 Palindrome Reorder/CSES Palindrome Reorder/d.out", "w", stdout);
     auto start = high_resolution_clock::now();

#endif

     ios_base::sync_with_stdio(false);
     cin.tie(NULL);

     string s;

     cin>>s;

     solve(s);

#ifndef ONLINE_JUDGE
     auto end = high_resolution_clock::now();
     std::chrono::duration<double> time=(end- start);
     milliseconds d=
      std::chrono::duration_cast<std::chrono::milliseconds>(time);
     cout<<"\n"<<d.count()<<"ms\n";
#endif

     return 0;
}
```