Couper les bûches: solutions

Lang: C++
knowledge requirement: vectors, strings
running time: O(L)

```
#include <bits/stdc++.h>

using namespace std;


int main() {
  ios::sync_with_stdio(false);
  int L;
  cin >> L;

  string s;
  cin >> s;

  vector<string> goods;

  int start = 0;
  int end = 0;

  while (end < L) {
    while (s[end] == 'O')
      end++;

    string good = s.substr(start, end - start);
    if (good.size() != 0)
      goods.push_back(good);

    while (s[end] == 'X')
      end++;

    start = end;
  }

  cout << goods.size() << '\n';
  for (auto g: goods)
    cout << g << '\n';

  return 0;
}
```

Lang: C++
knowledge requirement: vectors, strings functions (strdup, strtok)
running time: O(L)

```
#include <bits/stdc++.h>

using namespace std;

void split(const string &s, const char* delim, vector<string> & v){
  // to avoid modifying original string
  // first duplicate the original string and return a char pointer then free the
memory
  char *dup = strdup(s.c_str());
  char *token = strtok(dup, delim);
  while(token != NULL){
    v.push_back(string(token));
    // the call is treated as a subsequent calls to strtok:
    // the function continues from where it left in previous invocation
    token = strtok(NULL, delim);
  }
  free(dup);
}

int main() {
  ios::sync_with_stdio(false);
  int L;
  cin >> L;

  string s;
  cin >> s;

  vector<string> goods;

  split(s,"X",goods);

  cout << goods.size() << '\n';
  for (auto g: goods)
  if (!g.empty())
    cout << g << '\n';

  return 0;
}
```

Lang: C++ (ITA's solution)
knowledge requirement: vectors, strings
running time: O(L)

```
#include <iostream>
#include <bits/stdc++.h>

using namespace std;

string buche;
vector<string> res;
int main()
{
    int l;
  cin>>l;
  cin>>buche;

  for(int i=0;i<l;i++) {
    string curr="";
    while(buche[i]=='O') {
        curr+='O';
        i++;
    }
    if(curr.size()!=0) {
        res.push_back(curr);
    }
  }
  cout<<res.size()<<'\n';
  for(int i=0;i<res.size();i++) {
    cout<<res[i]<<'\n';
  }

}
```