Cut a string into a minimal number of substrings that are all palindromes.

bacacababbbabbababab → bacacab | abbba | b | bababab

abcdefghijklmno → a | b | c | d | e | f | g | h | i | j | k | l | m | n | o

```
$ a.out bacacababbbabbababab
 1 0 0 0 0 0 7 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 1 0 3 0 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 1 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 1 0 3 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 1 0 3 0 0 0 7 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 1 0 0 0 5 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 1 2 3 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 1 2 0 0 5 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 1 0 3 0 0 6 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 1 0 0 4 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 1 2 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 3 0 5 0 7
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 3 0 5 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 3 0 5
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 3 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 3
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1

 0 1 2 1 2 1 0 1 2 2 2 1 2 2 3 3 3 3 3 3

3 cuts are needed

bacacab|abbba|b|bababab
```

```
#include <iostream>
#include <iomanip>
#include <string>
#include <cstring>

using namespace std;

int main(int argc, char * argv[])
{ string s = argv[1];
  int n = s.length();
  int * * palen = new int * [n];
  for (int i = 0; i < n; i += 1)
  { palen[i] = new int[n];
    memset(palen[i], 0, n * sizeof(int));
    palen[i][i] = 1; }

  for (int i = 0; i < n - 1; i += 1)
    if (s[i] == s[i + 1])
      palen[i][i + 1] = 2;

  for (int len = 3; len <= n; len += 1)
    for (int start = 0; start <= n - len; start += 1)
    { int end = start + len - 1;
      if (s[start] == s[end] && palen[start + 1][end - 1] > 0)
        palen[start][end] = palen[start + 1][end - 1] + 2; }

  for (int i = 0; i < n; i += 1)
  { for (int j = 0; j < n; j += 1)
      cout << setw(2) << palen[i][j];
    cout << "\n"; }

  int * leastendingat = new int[n];
  leastendingat[0] = 0;

  for (int cutpos = 0; cutpos < n; cutpos += 1)
  { if (palen[0][cutpos] > 0)
    { leastendingat[cutpos] = 0;
      continue; }
    int least = 0x7FFFFFFF;
    for (int start = cutpos; start > 0; start -= 1)
      if (palen[start][cutpos] > 0 &&
          leastendingat[start - 1] + 1 < least)
        least = leastendingat[start - 1] + 1;
    leastendingat[cutpos] = least; }

  cout << "\n";
  for (int i = 0; i < n; i += 1)
    cout << setw(2) << leastendingat[i];
  cout << "\n\n";
```

```
cout << leastendingat[n - 1] << " cuts are needed\n\n";
string * subs = new string[leastendingat[n - 1] + 1];
int here = n - 1, next = here - 1;
while (leastendingat[here] > 0)
{ while (leastendingat[next] != leastendingat[here] - 1 ||
         palen[next + 1][here] == 0)
    next -= 1;
  subs[leastendingat[here]] = s.substr(next + 1, here - next);
  here = next; }
subs[0] = s.substr(0, here + 1);

for (int i = 0; i <= leastendingat[n - 1]; i += 1)
{ if (i > 0)
    cout << "|";
  cout << subs[i]; }
cout << "\n"; }
```