Return previous element in an expanding matrix

We have a square matrix whose size is continuously expanding by factor of 2. Given a sequence present in the matrix at position (i, j) at any point of time, we need to return sequence present at position (i, (j + N -1)%N) where N is size of the matrix.

When we say the matrix is expanding, the expanded matrix is formed by multiplying each element of the original 2 x 2 matrix with current N x N matrix itself. The expanded matrix will have dimensions 2N x 2N.

```
For Instance, consider below 2x2 matrix,
  [a b]
  [c d]

Expanding it will result in a 4x4 matrix as follows:
ax[a b]  bx[a b]      [aa ab ba bb]
  [c d]    [c d]      [ac ad bc bd]
                  --> [ca cb da db]
cx[a b]  dx[a b]      [cc cd dc dd]
  [c d]    [c d]

Expanding it again results in an 8x8 matrix as follows, and so on.

ax[aa ab ba bb]   bx[aa ab ba bb]     [aaa aab aba abb baa bab bba bbb]
  [ac ad bc bd]     [ac ad bc bd]     [aac aad abc abd bac bad bbc bbd]
  [ca cb da db]     [ca cb da db]     [aca acb ada adb bca bcb bda bdb]
  [cc cd dc dd]     [cc cd dc dd]     [acc acd adc add bcc bcd bdc bdd]
                                  --> [caa cab cba cbb daa dab dba dbb]
cx[aa ab ba bb]   dx[aa ab ba bb]     [cac cad cbc cbd dac dad dbc dbd]
  [ac ad bc bd]     [ac ad bc bd]     [cca ccb cda cdb dca dcb dda ddb]
  [ca cb da db]     [ca cb da db]     [ccc ccd cdc cdd dcc dcd ddc ddd]
  [cc cd dc dd]     [cc cd dc dd]
```

Basically for a given sequence, we need to find out the sequence just left to it. The matrix may be assumed circular i.e. sequence present at postion (i, 0) should return sequence present at position (i, N-1)

Examples:

```
Input: str = dda
Output: dcb

Input: str = cca
Output: ddb

Input: str = aacbddc
Output: aacbdcd
```

We strongly recommend you to minimize your browser and try this yourself first.

If we carefully analyze, we can see a pattern here.

Algorithm:
We start scanning the string from rightmost position and for each character do the following –

1. If the current character is ‘b’ or ‘d’, change to ‘a’ or ‘c’ respectively and return the string.

2. If the current character is ‘a’ or ‘c’, change it to ‘b’ or ‘d’ respectively and move to the next character to the left. Repeat Step 1 for the next left character.

```
// Program to return previous element in an expanding
// matrix.
#include <bits/stdc++.h>
using namespace std;

// Returns left of str in an expanding matrix of
// a, b, c  and d.
string findLeft(string str)
{
    int n = str.length();

    // Start from rightmost position
    while (n--)
    {
        // If the current character is ‘b’ or ‘d’,
        // change to ‘a’ or ‘c’ respectively and
        // break the loop
        if (str[n] == 'd')
        {
            str[n] = 'c';
            break;
        }
        if (str[n] == 'b')
        {
            str[n] = 'a';
            break;
        }

        // If the current character is ‘a’ or ‘c’,
        // change it to ‘b’ or ‘d’ respectively
        if (str[n] == 'a')
            str[n] = 'b';
        else if (str[n] == 'c')
            str[n] = 'd';
    }

    return str;
}

// driver program to test above method
int main()
{
    string str = "aacbddc";
    cout << "Left of " << str << " is "
         << findLeft(str);
   return 0;
}
```

Output :

```
Left of aacbddc is aacbdcd
```