Minimal Rotation:

```cpp
// the lexicographically minimal rotation of a string
// the rotation is shift left
// acab → caba → abac → baca
int minimal_rotation(string s) {
    int n = s.length();
    vector<int> f(2 * n, -1);
    int k = 0;
    for (int j = 1; j < 2 * n; j++) {
        int i = f[j - k - 1];
        while (i != -1 && s[j % n] != s[(k + i + 1) % n]) {
            if (s[j % n] < s[(k + i + 1) % n]) {
                k = j - i - 1;
            }
            i = f[i];
        }
        if (i == -1 && s[j % n] != s[(k + i + 1) % n]) {
            if (s[j % n] < s[(k + i + 1) % n]) {
                k = j;
            }
            f[j - k] = -1;
        } else {
            f[j - k] = i + 1;
        }
    }
    return k;
}
```

Longest Regular Bracket substring in string in O(n):

```cpp
string s;
int dp[1000100]; //fill with zero

int LRBSubstring(){
    int ans = 0;
    for (int i = 1; i < s.size(); i++)
    {
        if(s[i]==')'){
            if(s[i-1]=='(')
                dp[i] = (i >= 2 ? dp[i-2] : 0) + 2;
            else if(i-dp[i-1] > 0 && s[i-dp[i-1]-1]=='(')
                dp[i] = dp[i-1] + (i-dp[i-1] >= 2 ? dp[i - dp[i-1] - 2] : 0) + 2;
            ans = max(ans , dp[i]);
        }
    }
    return ans;
}
```