Algorithm 1:non- recursive

#include <stdio.h>

#include <string.h>

int is\_balanced(char\* substr)

{

    int count[2] = {0};

    for(int i = 0; i < strlen(substr); i++)

    {

        if(substr[i] == substr[0])

            count[0]++;

        else

            count[1]++;

    }

    return (count[0] == count[1]);

}

int longest\_balanced\_substring(char\* s)

{

    int n = strlen(s);

    int max\_length = 0;

    for(int i = 0; i < n-1; i++)

    {

        for(int j = i+2; j <= n; j++)

        {

            char substr[j-i];

            strncpy(substr, &s[i], j-i);

            substr[j-i] = '\0';

            if(is\_balanced(substr) && strlen(substr) > max\_length)

                max\_length = strlen(substr);

        }

    }

    return max\_length;

}

int main()

{

    char s1[] = "cabbacc";

    printf("the longest balanced substring for ( %s ) = %d\n",s1,longest\_balanced\_substring(s1)); *// Expected Output: 4*

    char s2[] = "abababa";

    printf("the longest balanced substring for ( %s ) = %d\n",s2,longest\_balanced\_substring(s2)); *// Expected Output: 6*

    char s3[] = "aaaaaaa";

    printf("the longest balanced substring for ( %s ) = %d\n",s3,longest\_balanced\_substring(s3)); *// Expected Output: 0*

    return 0;

}

Algorithm 2: recursive

#include <stdio.h>

#include <string.h>

int is\_balanced(int\* freq) {

    int i;

    int count = 0;

    int diff\_chars = 0;

    for (i = 0; i < 26; i++) {

        if (freq[i] > 0) diff\_chars++;

        if (freq[i] == freq[0] && freq[i] > 0) count++;

    }

    if (diff\_chars == 2 && count == 2) return 1;

    return 0;

}

int longest\_balanced\_substring\_helper(char\* s, int start, int end) {

    if (end - start + 1 < 2) {

        return 0; *// substring too short to be balanced*

    }

    int freq[26] = {0};

    int i;

    for (i = start; i <= end; i++) {

        freq[s[i] - 'a']++;

    }

    if (is\_balanced(freq)) {

        return end - start + 1; *// this substring is balanced*

    }

    int len1 = longest\_balanced\_substring\_helper(s, start, end - 1);

    int len2 = longest\_balanced\_substring\_helper(s, start + 1, end);

    return (len1 > len2) ? len1 : len2; *// return the longer of the two balanced substrings*

}

int longest\_balanced\_substring(char\* s) {

    return longest\_balanced\_substring\_helper(s, 0, strlen(s) - 1);

}

int main1() {

    char s1[] = "cabbacc";

    printf("the longest balanced substring for ( %s ) = %d\n",s1 ,longest\_balanced\_substring(s1)); *// Expected Output: 4*

    char s2[] = "abababa";

    printf("the longest balanced substring for ( %s ) = %d\n",s2 ,longest\_balanced\_substring(s2)); *// Expected Output: 6*

    char s3[] = "aaaaaaa";

    printf("the longest balanced substring for ( %s ) = %d\n",s3 ,longest\_balanced\_substring(s3)); *// Expected Output: 0*

    return 0;

}