```c
int isPalindrome(char * s, int l, int r)
{
    int len;

    if(l<=r)
    {
      len = 0;
    }else
    {
        len = -1;
    }

    while(l<=r)
    {
        if(s[l] != s[r])
        {
            len = -1;
            break;
        }

        if(l == r)
        {
          len++;
        }else
        {
          len+=2;
        }

        l++;
        r--;
    }

    return len;
}


char * longestPalindrome(char * s){
    int l_index;
    int r_index;
    int str_len;
    int max_len;
    int max_l;
    int max_r;
    int len;


    str_len = strlen(s);
    l_index = 0;
    max_len = 0;
    max_l = 0;
    max_r = 0;

    for(l_index = 0 ; l_index < str_len; l_index++)
    {
        for(r_index = str_len - 1; r_index >= l_index;r_index--)
        {
            if( s[l_index] == s[r_index] )
            {
                len = isPalindrome(s, l_index, r_index);
                if(len > max_len)
                {
                    max_len = r_index - l_index + 1;
                    max_l = l_index;
                    max_r = r_index;
                }
            }
        }
    }

    s[max_r+1] = '\0';

    return  &s[max_l];
}
```