Lecture-22

All about Char Arrays, Strings & solving LeetCode Questions

Strings vs char arrays--> https://techdifferences.com/difference-between-character-array-and-string.html

Q1. Length of a string

```
#include<iostream>
using namespace std;
int main(){
char name[20];
cin>>name;
int count=0;
for(int i=0;name[i]!='\0';i++){
count++;
}
cout<<"Length is "<<count;
}
```

Q2. Reverse a string (344)
https://leetcode.com/problems/reverse-string/

```
class Solution {
public:
void reverseString(vector<char>& s) {
int st=0;
int e=s.size()-1;
while(st<e){
swap(s[st],s[e]);
st++;
e--;
}
}
};
```

Q3. Check Palindrome

https://www.codingninjas.com/studio/problems/check-if-the-string-is-a-palindrome_1062633

```
#include <bits/stdc++.h>
bool isValid(char ch){
if((ch>='a' && ch<='z') || (ch>='A' && ch<='Z') || (ch>='0' && ch<='9')){
return true;
}
return false;
}
char toLower(char ch){
char temp;
if((ch>='a' && ch<='z') || (ch>='0' && ch<='9')){
return ch;
}
else if(ch>='A' && ch<='Z'){
temp = ch - 'A' + 'a';
return temp;
}else{
return ch;
}
}
bool validPalindrome(string s){
int st=0, end=s.length()-1;
while(st<=end){
if(s[st]==s[end]){
st++;
end--;
}else{
return false;
}
}
return true;
}
bool checkPalindrome(string s)
{
// Write your code here.
string temp;
for(int j=0;j<s.length();j++){
if(isValid(s[j])){
temp.push_back(s[j]);
}
}
for(int j=0;j<temp.length();j++){
temp[j] = toLower(temp[j]);
}
```

```
return validPalindrome(temp);
}
```

Q4. Valid Palindrome (125)
https://leetcode.com/problems/valid-palindrome/
Same as above

Q5. Reverse Words

Q6. maximum occurring character
https://practice.geeksforgeeks.org/problems/maximum-occuring-character-1587115620/1

```
char getMaxOccuringChar(string str)
{
// Your code here
int max=0;
char ch, newch;
for(int i=0;i<str.length();i++){
int count=0;
for(int j=0;j<str.length();j++){
if(str[i]==str[j]){
count++;
}
}
if(count>max){
max=count;
ch = str[i];
}else if(count==max){
if(str[i]<ch){
ch=str[i];
}
}
}
return ch;
}
```

Q7. Replace words
https://www.codingninjas.com/studio/problems/replace_spaces_1172172

```
#include <bits/stdc++.h>
```

```
string replaceSpaces(string &str){
// Write your code here.
string temp="";
for(int i=0;i<str.length();i++){
if(str[i]==' '){
temp.push_back('@');
temp.push_back('4');
temp.push_back('0');
}else{
temp.push_back(str[i]);
}
}
return temp;
}
```

Q8. 1910. Remove All Occurrences of a Substring

```
class Solution {
public:
string removeOccurrences(string s, string part) {
while(s.length()!=0 && s.find(part)<s.length()){
s.erase(s.find(part), part.length());
}
return s;
}
};
```

Q9. 567. Permutation in String

```
class Solution {
private:
bool checkEqual(int a[26], int b[26]){
for(int i=0;i<26;i++){
if(a[i]==b[i]){
continue;
}else{
return false;
}
}
return true;
}
public:
bool checkInclusion(string s1, string s2) {
//character count array
int count[26] = {0};
```

```
int index = s1[i] - 'a';
count[index]++;
}
int i=0;
int windowSize = s1.length();
int count2[26] = {0};
//for first window
while(i<windowSize && i<s2.length()){
int index = s2[i] - 'a';
count2[index]++;
i++;
}
if(checkEqual(count, count2)){
return 1;
}
//for next windows
while(i<s2.length()){
char newChar = s2[i];
int index=s2[i] - 'a';
count2[index]++;
char oldChar = s2[i-windowSize];
index = oldChar - 'a';
count2[index]--;
i++;
if(checkEqual(count, count2)){
return 1;
}
}
return 0;
}
};
```

Q10. [1047. Remove All Adjacent Duplicates In String]

My solution giving TLE

```
class Solution {
public:
string removeDuplicates(string s) {
int i=0,j=1;
while(i<s.length() && j<s.length()){
int j=i+1;
```

```
if(s[i]==s[j]){
while(s[i]==s[j]){
j++;
}
int length = j-i;
if(length%2==0){
s.erase(i,length);
}else{
s.erase(i+1, length-1);
}
i=0;
j=1;
}else{
i++;
}
}
return s;
}
};
```

Soln:

```
class Solution {
public:
string removeDuplicates(string s)
{
int i = 0;
int j = 1;
while (j < s.length())
{
if (s[i] == s[j])
{
s.erase(i, 2);
i = max(0, i - 1);
j = max(1, j - 1);
}
else
{
i++;
j++;
}
}
return s;
}
};
```

Q11. 443. String Compression

```
class Solution {
public:
int compress(vector<char>& chars) {
int i=0;
int ansIndex=0;
int n=chars.size();
while(i<n){
int j=i+1;
while(j<n && chars[j]==chars[i]){
j++;
}
chars[ansIndex++] = chars[i];
int count = j-i;
if(count>1){
string cnt = to_string(count);
for(char ch: cnt){
chars[ansIndex++] = ch;
}
}
i=j;
}
return ansIndex;
}
};
```