```c
#define ALLOC_LENGTH            (100)
char * countAndSay(int n){
    char current;
    char* tmp[2];
    int index;
    int count;
    int str_index;
    int next_str_index;
    char num;
    int alloc_length;

    alloc_length = ALLOC_LENGTH;
    tmp[0] = (char*)malloc( sizeof(char)*alloc_length );
    tmp[1] = (char*)malloc( sizeof(char)*alloc_length );

    tmp[1][0] = '1';
    tmp[1][1] = '\0';

    count = 0;

    for(index = 2; index <= n; index++)
    {
        str_index = 0;
        next_str_index = 0;
        num = '*';
        while(true)
        {
            if(num == '*')
            {
                num = tmp[(index-1)%2][str_index];
            }else if(num != tmp[(index-1)%2][str_index])
            {
                next_str_index++;
                tmp[index%2][next_str_index] = num;
                next_str_index++;
                if( (next_str_index+2) % alloc_length == 0)
                {
                   alloc_length += ALLOC_LENGTH;
                   tmp[0] = (char*)realloc(tmp[0], sizeof(char)*alloc_length );
                   tmp[1] = (char*)realloc(tmp[1], sizeof(char)*alloc_length );
                }
                num = tmp[(index-1)%2][str_index];
                count = 0;
            }

            if(tmp[(index-1)%2][str_index] == '\0')
            {
                break;
            }

            count++;
            tmp[index%2][next_str_index] = count + '0';

            str_index++;
        }

        tmp[index%2][next_str_index] = '\0';

    }

    free(&tmp[(n+1)%2][0]);

    return &tmp[n%2][0];
}
```