第二周习题刷题记录

14.最长公共前缀

题目：编写一个函数来查找一个字符串数组中的最小公共前缀。如果不存在公共前缀，则返回空字符串“”。

算法：

1. 暴力枚举
    - 取出第一个单词，并依次比较字母与后面单词的字母是否相同
    - 时间复杂度为$O(s)$,其中s为所有字符串的单词之和

```
char * longestCommonPrefix(char ** strs, int strsSize){
    if(strsSize == 0 || strs[0] == ""){
        return "";
    }
    for(int i = 0; i < strlen(strs[0]); i++){
        for(int j = 1; j < strsSize; j++){
            if(strs[0][i] != strs[j][i]){
                strs[0][i] = '\0';
                break;
            }
        }
    }
    return strs[0];
}
```

300 .最长上升子序列

题目：给出一个未排序的整数数组，找出最长递增子序列的长度

算法：

1. 动态规划
    - 用数组$dp[i]$ 记录以$nums[i]$ 结尾（即$nums[i]$ 为最后一个数字）的最长递增子序列的长度，则递推方程为$dp[i] = max(dp[i], dp[j] + 1)$，其中要求$0 \leq j < i$且$nums[j] < nums[i]$。
    - 其时间复杂度需要进行两层遍历，因此需要$O(n^2)$，空间复杂度需要一个额外的dp 数组，因此空间复杂度为$O(n)$。

```
int max(int a, int b){
    return a > b ? a : b;
}

int lengthOfLIS(int* nums, int numsSize){
    int res = 0;
    if(numsSize == 0){
        return res;
    }
    int *dp = (int *)malloc(numsSize * sizeof(int));
    for(int i = 0; i < numsSize; i++){
        dp[i] = 1;
    }
```

```
    for(int i = 0; i < numsSize; i++){
        for(int j = 0; j < i; j++){
            if(nums[j] < nums[i]){
                dp[i] = max(dp[i], dp[j] + 1);
            }
        }
    }
    for(int i = 0; i < numsSize; i++){
        res = max(res,dp[i]);
    }
    return res;
}
```