华为实验班作业（20200919）

一、代码实践（自己整理的leetcode题目及总结）

1、leetcode46 全排列

本题考点：递归&回溯
解题思路：定义递归函数backtrack（first，output）：表示遍历到第first个位置时，当前的排列为output
则有以下几种情况：
（1）当first==n时，则遍历结束，将当前的output并入最后的结果res中，递归结束
（2）当first< n时，填数，比较初级的想法是设置一个vistited数组，以标记访问过的数，但可以利用数组进行分割，将数组分成左右两边，分别为已访问的和未访问的，在访问时将当前值与已访问区的边界值交换即可
需要注意的是，根据回溯算法，在完成对当前数的访问之后，需要通过交换与之前的边界值交换回来，从而实现回退
关键步骤：
（1）主函数需要建立一个包含全部数字的列表
（2）之后调用辅助函数backtrack进行回溯操作
（3）backtrack函数中首先对序号进行判断是否结束递归
（4）若不是，则将当前值与已访问区的边界进行交换，之后进行下层递归，递归结束后交换回来实现回退
//leetcode官方题解视频里有使用used数组的代码

```
class Solution {
    public List<List<Integer>> permute(int[] nums) {
        List<List<Integer>> res = new ArrayList<List<Integer>>();
        List<Integer> output = new ArrayList<Integer>();

        for(int num : nums){
            output.add(num);
        }

        int n = nums.length;
        backtrack(n,output,res,0);
        return res;
    }

    public void backtrack(int n,List<Integer> output,List<List<Integer>> res,int
first){
        if(first == n){
            res.add(new ArrayList<Integer>(output)); //复制链表
        }
        for(int i=first;i<n;i++){
            Collections.swap(output,first,i);
            backtrack(n,output,res,first+1);
            Collections.swap(output,first,i);
        }
    }
}
```

2、leetcode47 全排列II

关键在于不重复
这里先对原始数组进行sort排序，保证重复的元素一定相邻，而后在backtrack中加入元素时增加一步判断即可
`if (vis[i] || (i > 0 && nums[i] == nums[i - 1] && !vis[i - 1]))`

```
class Solution {
    boolean[] vis;

    public List<List<Integer>> permuteUnique(int[] nums) {
        List<List<Integer>> ans = new ArrayList<List<Integer>>();
        List<Integer> perm = new ArrayList<Integer>();
        vis = new boolean[nums.length];
        Arrays.sort(nums);
        backtrack(nums, ans, 0, perm);
        return ans;
    }

    public void backtrack(int[] nums, List<List<Integer>> ans, int idx,
List<Integer> perm) {
        if (idx == nums.length) {
            ans.add(new ArrayList<Integer>(perm));
            return;
        }
        for (int i = 0; i < nums.length; ++i) {
            if (vis[i] || (i > 0 && nums[i] == nums[i - 1] && !vis[i - 1])) {
                continue;
            }
            perm.add(nums[i]);
            vis[i] = true;
            backtrack(nums, ans, idx + 1, perm);
            vis[i] = false;
            perm.remove(idx);
        }
    }
}
```

二、计算机基础知识整理

1、Ipv4转Ipv6的三种方案

（1）双栈技术：主机或路由器同时装有IPV4 和 IPV6两个协议栈，因此，主机既能和IPV4通信，也能和IPv6网络通信。

备注：目前市面上很多路由器都是双栈路由器，即同时可以访问ipv4与ipv6

（2）隧道技术：在IPV6分组进入IPV4网络时，将IPV6分组封装成IPV4分组；当封装成IPV4分组离开IPV4网络时，再装数据部分（IPV6部分）转发给目的节点。

用隧道技术可以通过现有的运行IPv4协议的Internet骨干网络（即隧道）将局部的IPv6网络连接起来，因而是IPv4向IPv6过渡的初期最易于采用的技术。路由器将IPv6的数据分组封装入IPv4，IPv4分组的源地址和目的地址分别是隧道入口和出口的IPv4地址。在隧道的出口处，再将IPv6分组取出转发给目的站点

> 备注：这是早期的通信方式，使用较多的有清华开源的ISATAP隧道，但目前已经不再维护
> 该项目地址为：https://github.com/tuna/ipv6.tsinghua.edu.cn/blob/master/isatap.md?spm=a2c4e.10696291.0.0.5fe319a4urvpNR&file=isatap.md

（3）协议翻译技术：对IPV6和IPV4报头时行相互翻译，实现IPV4/IPV6协议和地址的转换。
网络地址转换/协议转换技术 NAT-PT 通过与SIIT协议转换和传统的IPv4下的动态地址翻译（NAT）以及适当的应用层网关（ALG）相结合，实现了只安装了IPv6的主机和只安装了IPv4机器的大部分应用的相互通信

> 备注：还没有对这种方式进行了解，将在之后的学习中展开

2、http和https区别

http运行在TCP之上。所有传输的内容都是明文，客户端和服务器端都无法验证对方的身份。
https是HTTP运行在SSL/TLS之上，SSL/TLS运行在TCP之上。所有传输的内容都经过加密（对称加密）。此外客户端可以验证服务器端的身份，如果配置了客户端验证，服务器也可以验证客户端的身份

三、小组项目工作总结

1、开源训练营相关

学习了开源社区中issue相关的内容：

（1）方便了沟通，方便了检索，有利于项目维护

（2）可以在issue中描述的问题：未解决的问题、程序的bug、其他项目问题的交流

（3）在issue提交过程中的礼仪：“开源社区互惠互利，谁也不欠谁”，提问之前先检索相关历史，内容尽可能充分（bug说明运行配置、最好能提供日志或最小复现仓库）

（4）相关文件（模板化）：

CODE_OF_CONDUCT：沟通行为准则

CONTRIBUTING.md：提供有效的沟通指引（说明如何提交issue等）

ISSUE_TEMPLATE.md：提供提交issue的模板

2、小组项目相关

讨论了目前项目的进展，并制定了接下来项目维护开发的计划

（1）重新制定项目的两大分支板块，并确立了各部分的负责人

（2）规划了未来项目的发展方向，并将之前的前端等项目代码加入