2022 年腾讯精选面试题及答案

1. 删除字符串 s1 中在字符串 s2 中出现的字符。

基本思路:把 s1 的字符存到一个 set 里面,然后遍历 s2,看是否出现过,出现过就 erase 掉。但是直接输出 set 的元素这样会改变顺序，要想顺序不变，就顺序遍历一下 s1 看是否出现，出现就输出。

```
#include <iostream>
#include <cstdio>
#include <cstring>
#include <cmath>
#include <algorithm>
#include <vector>
#include <set>
#include <queue>
#include <map>
using namespace std;
typedef long long LL;
const int maxn=1005;
set<char>s;
int main()
{
    string s1,s2;
    cin>>s1>>s2;
    int len=s1.length();
    for (int i=0;i<len;i++)
        s.insert(s1[i]);
    len=s2.length();
    for (int i=0;i<len;i++)
    {
        if (s.count(s2[i]))
        s.erase(s.find(s2[i]));
    }
    len=s1.length();
    for (int i=0;i<len;i++)
    {
        if (s.count(s1[i]))
            cout<<s1[i];
    }
    cout<<endl;
    return 0;
```

```
}
```

2. 求一个论坛的在线人数，假设有一个论坛，其注册 ID 有两亿个,每个 ID 从登陆到退出会向一个日志文件中记下登陆时间和退出时间，要求写一个算法统计一天中论坛的用户在线分布，取样粒度为秒。

一天总共有 3600*24=86400 秒。
定义一个长度为 86400 的整数数组 intdelta[86400]，每个整数对应这一秒的人数变化值，可能为正也可能为负。开始时将数组元素都初始化为 0。
然后依次读入每个用户的登录时间和退出时间，将与登录时间对应的整数值加 1，将与退出时间对应的整数值减 1。
这样处理一遍后数组中存储了每秒中的人数变化情况。
定义另外一个长度为 86400 的整数数组 intonline_num[86400]，每个整数对应这一秒的论坛在线人数。
假设一天开始时论坛在线人数为 0，则第 1 秒的人数 online_num[0]=delta[0]。第 n+1 秒的人数 online_num[n]=online_num[n-1]+delta[n]。
这样我们就获得了一天中任意时间的在线人数。

3. 有序链表合并.

```
/**
* Definition for singly-linked list.
* struct ListNode {
* int val;
* ListNode *next;
* ListNode(int x) : val(x), next(NULL) {}
* };
*/
class Solution {
    public:
    ListNode* mergeTwoLists(ListNode* l1, ListNode* l2) {
        if (l1 == NULL) {
            return l2;
        } else if (l2 == NULL) {
            return l1;
        } else {
            if (l1->val <= l2->val) {
```

```
                l1->next = mergeTwoLists(l1->next, l2);
                return l1;
            } else {
                l2->next = mergeTwoLists(l1, l2->next);
                return l2;
            }
        }
    }
};
```

4. 有 n 种硬币，面额分别为 1~n，每种硬币都有无限个，假设要付款的金额为 m。

m/n+!!(m%n)

5. 一个数列：-1 2 -3 4 -5 6 ... 询问 q 次，每次询问区间[l,r]的区间和,输出每个询问的答案。

第 1 个和第 2 个加起来为 1，第 3，4 个加起来也为 1.....
所以前 i 项和为：
i/2+(i&1)*i;
区间和可以用前 i 项和算出来了

6. 牛妹有剪刀，石头，布（以 0，1，2 表示）三种卡片无限张。现在牛妹拿出 n 张排成一排。然后你也拿出 n 张牌一一对应比对。若赢一局则获得一分。若你想得 k 分。现在输入 n，k 和牛妹的 n 张牌分别是什么，你想要恰好得 k 分，有多少种方法。

很容易想到答案跟牛妹每一张牌是什么没有关系。没一张牌只需要考虑赢、不赢。
赢 k 分，那就是从 n 张牌中拿出 k 张赢，
其他输，所以组合数 c(n,k).对于赢了答 k 张，只有一种方法，但是对于剩下的 n-k 张，都有平局和输掉两种情况，所以是 2 的 n-k 次方。

两者相乘就是答案。
结果很大对 mod=1e9+7 取余，用到同余定理。
求 2 的幂直接暴力求（当然也可以快速幂）
求组合数的时候用到除法，
又要取余，所以用到逆元。所以用到逆元公式（当然还有其他求法）:pow(x,mod-2)%mod;
但是 mod=1e9+7，所以暴力求幂会超时，
方法是用快速求幂法压缩时间（快速幂就不贴代码了）

```
typedef long long ll;
ll fast(ll a,ll n) // 快速幂 pow(a,n)
ll inv(ll x, ll mod)
{
return fast(x,mod-2);
}
```

7. const 的含义及实现机制,比如: const int 1,是怎么做到 i 只可读的?

const 用来说明所定义的变量是只读的。
这些在编译期间完成,编译器可能使用常数直接替换掉对此变量的引用。

8. 有一个射击游戏有 m 种颜色的气球，颜色分别为 1~m 现在一个人开了 n 枪，告诉你一个数列，表示打爆的气球颜色分别是多少。（注意，0 表示这一枪没有打中，mmp 这里害得我 debug 了好久）求一个最小区间[l,r]，在区间内包含了所有 1~m 颜色。输出区间长度。

这个题是 XUPT 2019 寒假训练最后一场比赛的原题的强化版。刚好我做了并且在 bilibli 上给学弟学妹们讲了，很奈斯。
用一个变量维护当前区间里有多少种颜色，用 book 数组表示第 i 种颜色在当前区间内出现了多少次。
然后尺取。

9. 到商店里买 200 的商品返还 100 优惠券(可以在本商店代替现金)。请问实际上折扣是多少?

由于优惠券可以代替现金,所以可以使用200元优惠券买东西,然后还可以获得100元的优惠券。
假设开始时花了 x 元,那么可以买到 x+x/2+x/4+.的东西。所以实际上折扣是 50%(当然,大部分
时候很难一直兑换下去,所以 50%是折扣的上限)
如果使用优惠券买东西不能获得新的优惠券,那么总过花去了 200 元,可以买到200+100元的商品,所以实际折扣为 200/300=67%。

10. TCP 三次握手的过程,accept 发生在三次握手哪个阶段?

accept 发生在三次握手之后。
第一次握手:客户端发送 syn 包(syn=j)到服务器。
第二次握手:服务器收到syn包,必须确认客户的sY(ack=j+1),同时自己也发送一个ASK包(ask=k)。
第三次握手:客户端收到服务器的 SYN+ACK 包,向服务器发送确认包 ACK(ack=k+1)。
握手完成后,客户端和服务器就建立了 tcp 连接。这时可以调用 accept 函数获得此连接。

11. 用 UDP 协议道讯时怎样得知目标机是否获得了数据包 ?

可以在每个数据包中插入一个唯一的 ID,比如 timestamp 或者递增的 int。
发送方在发送数据时将此 ID 和发送时间记录在本地。
接收方在收到数据后将 ID 再发给发送方作为回应。
发送方如果收到回应,则知道接收方已经收到相应的数据包;如果在指定时间内没有收到回应,则数据包可能丢失,需要重复上面的过程重新发送一次,直到确定对方收到。

12. 求一个论坛的在线人数,假设有一个论坛,其注册 ID 有两亿个,每个 ID 从登陆到退出会向一个日志文件中记下登陆时间和退出时间,要求写一个算法统计一天中论坛的用户在线分布,取样粒度为秒。

一天总共有 3600*24=8600 秒。
定义一个长度为 86400 的整数数组 int delta[86400],每个整数对应这一秒的人数变化值,可能为正也可能为负。开始时将数组元素都初始化为 0。
然后依次读入每个用户的登录时间和退出时间,将与登录时间对应的整数值加 1,将与退出时间对应的整数值减 1。
这样处理一遍后数组中存储了每秒中的人数变化情况。
定义另外一个长度为 86400 的整数数组 int online num[86400,每个整数对应这一秒的论坛在线人数。
假设一天开始时论坛在线人数为 0,则第 1 秒的人数 online num[0]= delta[0]。第 n+1 秒的人数
line num[n]= online num[n-1]+delta[n]。
这样我们就获得了一天中任意时间的在线人数。

13. 从 10G 个数中找到中数在一个文件中有 10G 个整数,乱序排列,要求找出中位数。内存限制为 2G.

不妨假设 10G 个整数是 64bit 的。
2G 内存可以存放 256M 个 64bit 整数。
我们可以将 64bit 的整数空间平均分成 256M 个取值范围,用 2G 的内存对每个取值范围内出现整数个数进行统计。这样遍历一边 10G 整数后,我们便知道中数在那个范围内出现,以及这个范围内总共出现了多少个整数。
如果中数所在范围出现的整数比较少,我们就可以对这个范围内的整数进行排序,找到中数。如果这个范围还可以采用同样的方法将此范围再次分成多个更小的范围
(256M-228,所以最多需要 3 次就可以将此范围缩小到 1,也就找到了中数)

14. 两个整数集合 A 和 B,求其交集。

1.读取整数集合 A 中的整数,将读到的整数插入到 mp 中,并将对应的值设为 1。
2.读取整数集合 B 中的整数,如果该整数在 map 中并且值为 1,则将此数加入到交集当中,并将在 map 中的对应值改为 2。
通过更改 map 中的值,避免了将同样的值输出两次。

15. 找出 1 到 10w 中没有出现的两个数字有 1 到 10w 这 10w 个数,去除 2 个并打乱次序,如何找出那两个数?

申请 10w 个 bit 的空间,每个 bit 代表一个数字是否出现过。
开始时将这 10 个 bit 都初始化为 0,表示所有数字都没有出现过。
然后依次读入已经打乱循序的数字,并将对应的 bit 设为 1。
处理完所有数字后,根据为 0 的 bi 得出没有出现的数字。
首先计算 1 到 10w 的和,平方和。
然后计算给定数字的和,平方和。
两次的到的数字相减,可以得到这两个数字的和,平方和。
所以我们有
x + y = n
x^2 + y^2 = m
解方程可以得到 x 和 y 的值。

16. 有 1000 瓶水,其中有一瓶有毒,小白鼠只要尝一点带奇的水 24 小时后就会死亡,至少要多少只小白鼠才能在 24 小时时鉴别出那瓶水有毒?

最容易想到的就是用 1000 只小白鼠,每只喝一瓶。但显然这不是最好答案。
既然每只小白鼠喝一瓶不是最好答案,那就应该每只小白鼠喝多瓶。那每只应该喝多少瓶呢?
首先让我们换种问法,如果有 x 只小白鼠,那么 24 小时内可以从多少瓶水中找出那瓶有毒的?
由于每只小白鼠都只有死或者活这两种结果,所以 x 只小白鼠最大可以表示 2^x 种结果。
如果让每种结果都对应到某瓶水有毒,那么也就可以从 2^x 瓶水中找到有毒的那瓶水。
那如何来实现这种对应关系呢?
第一只小白鼠喝第 1 到 2^(x-1)瓶,第二只小白鼠喝第 1 到第 2^(x-2)和第 2^(x-1)+1 到第 2^(x-1)+2(x-2)瓶...以此类推。
回到此题,总过 1000 瓶水,所以需要最少 10 只小白鼠。

17. 根据上排的数填写下排的数,并满足要求.

根据上排给出十个数,在其下排填出对应的十个数,要求下排每个数都是上排对应位置的数在下排出现的次数。上排的数:0,1,2,3,4,5,6,7,8,9。

18. 给 40 亿个不重复的 unsigned int 的整数,没排过序的,然后再给几个数,如何快速判断这几个数是否在那 10 亿个数当中?

unsigned int 的取值范围是 0 到 2^32-1。我们可以申请连续的 232/8-512M 的内存,用每一个 bit 对应个 unsigned int 数字。首先将 51M 内存都初始化为 0,然后每处理一个数字就将其对应的 bit 设置为 1。当需要查询时,直接找到对应 bit,看其值是 0 还是 1 即可。

19. 1-20 的两个数把和告诉 A 积告诉 B,A 说不知道是多少,B 也说不知道,这时 A 说我知道了,B 接着说我也知道了, 问这两个数是多少?

2 和 3

20. 爸爸妈妈妹妹小强,至少两个人同一生肖的概率是多少?

1-12*11*10*9/12*12212*12=1-5596=4196

21. 计算 ab<<2.

运算符优先级:括号,下标,->和(.成员)最高;
单目的比双目的高;
算术双目的比其他双目的高;
位运算高于关系运算;
关系运算高于按位运算(与,或,异或);
按位运算高于逻辑运算;
目的只有一个条件运算,低于逻辑运算;
赋值运算仅比,(顺序运算)高;
在此题中,位左移<<” 优先级高于按位异或""所以 b 先左移两位(相当于乘以 4)再与 a 异或。
例如:当 a=6,b=4 时;则 a^b<<2=22

22. 如何输出源文件的标题和目前执行行的行数?

```
printf("The file name:%dn", __FILE__);
printf("The current line No: %d\n", __LINE__);
```

ANSI C 标准预定义宏;

```
__LINE__
__FILE__
__DATE__
__TIME__
```

__STDC__ 当要求程序严格遵循 ANSC 标准时该标识符被赋值为 1

__cplusplus__ 当编写 C+程序时该标识符被定义

23. a[3]4 哪个不能表示 a[1]1: *(&a[0][0]+5) *((a+1)+1) *(&a[1]+1) *(&a[0][0]+4)*(&a[1]+1).

a 是数组的首地址,a[1]就表示 a[1][0]地址了,不用再取地.

23. fun((exp1,exp2),(exp3,exp4,exp5))几个实参?

两个.

形式参数:在声明和定义函数时,写在函数名后的括号中的参数.

实参是调用参数中的变里,行参是被调用函数中的变量.

24. 希尔,冒泡,快速,插入哪个平均速度最快?

快速排序

快速排序、归并排序和基数排序在不同情况下都是最快最有用的

25. enum 的声明方式

```
enum 枚举类型名{
枚举常量 1,
枚举常里 2,
...
枚举常里 n
};
```

For example:
enum weekday i sunday, monday, tuesday, wednesday,thursday, friday, saturday};
enum weekday week_day; //week_day 就是一个枚举类型变量

26. 频繁的插入删除操作使用什么结构比较合适,链表还是数组?

链表

27. *p=NULL; *p= new charl[100]; sizeof(p)各为多少?

都为 4。因为都是指针类型,所占存储空间必然为 4

28. 顺序查找的平均时间?

(1+2+3+...+n)/n=(n+1)/2

29. for(i=0,sum=0;i<10;++i,sum+=i)的运行结果?

sum=55

30. 不能做 switch()的参数类型是?

switch 的参数不能为浮点型

31. 不使用其他变里,交换两个整型 a,b 的值?

x=x+y; y=x-y; x=x-y

32. 写出 foat x 与"零值"比较的 if 语句。

if(x>=0.000001 && x<=-0.000001)(x 不为 0 的比较)
float: 6 位精度
double: 16 位精度

33. 腾讯服务器每秒有 2W 个 QQ 号同时上线,找出 5min 内重新登入的 qq 号并打印出来。

如果空间足够大,可以定义一个大的数组 a[qq 号],初始为零然后.这个 qq 号登陆了就 a[qq 号]++
最后统计大于等于 2 的 QQ 号
这个用空间来代替时间
不成熟的想法
2w x 300s
所以用 6000.000 个桶。删除超时的算法后面说,所以平均桶的大小是 1
假设 qq 号码一共有 10^10 个,所以每个桶装的 q 号码是 10^10/(6*10~6)个,
这个是插入时候的最坏效率(插入同个桶的时候是顺序查找插入位置的)
qq 的节点结构和上面大家讨论的基本一样,增加一个指针指
向输出列表,后面说

```
struct QQstruct {
    num_type qqnum,
    timestamp last_logon_time,
    QQstruct *pre,
    QQstruct *next,
    OutPutlist *out //用于 free 节点的时候,顺便更新下输出列表
}
```

另外增加两个指针列表
第一个大小 300 的循环链表,自带一个指向 QQStruct 的域,循环存 300 秒内的 qq 指针。
时间一过就 fee 掉,所以保证所有桶占用的空间在 2wX30 以内.
第二个是输出列表,就是存放题目需要输出的节点。
如果登陆的用户,5 分钟内完全没有重复的话,每秒 free2w 个节点
不过在 free 的时候,要判断一下时间是不是真的超时,因为把节点入桶的时候,遇到重复的,
会更新一下最后登陆的时间。当然啦,这个时候,要把这个 q 号码放到需要输出的列表里面

34. 给一个奇数阶 N 幻方,填入数字 1,2,3.N^N,使得横竖斜方向上的和都相同.

```
#include<iostream>
#include<iomanip>
#include<cmath>
usingnamespace std;
int main()
{
    int n;
    cin>>n;
    int i;
    int **Matr = new int *[n]; //动态分配二维数组
    for(i=0;i<n;++i)
        Matr[i]=new int[n]: //动态分配二维
    数组
    //j=n/2 代表首行中间数作为起点,即 1 所在位置
    int j=n/2,num=1: //初始值
    i=0;
    while(num!=n*n+1) {
        //往右上角延升, 若超出则用%转移到左下角
        Matr [(i%n+n)%n][(j%n+n)%n]=num;
        //斜行的长度和 n 是相等的,超出则转至下一写信.
        if (num%n==0) {
            i++;
        } else {
            i--;
            j++;
        }
        num++;
    }
    for (i=0;i<n;i++) {
        for (j=0;j<n;++j) {
            cout << setw((int)log10(n*n)+4)<<Matr[i][j]; //格式控制
            cout <<endl<<endl;//格式控制
        }
    }
    for (i=0; i<n; ++i) {
        delete [] Matr[i];
    }
    return 1;
}
```

35. IP 地址的编码分为哪俩部分?

网络号和主机号。不过是要和子网掩码按位与上之后才能区分哪些是网络位哪些是主机位

36. 描述实时系统的基本特性.

在特定时间内完成特定的任务,实时性与可靠性

37. Internet 采用哪种网络协议?该协议的主要层次结构?

TCPP 协议。应用层、传输层、网络层、数据链路层和物理层

38. Internet 物理地址和|P 地址转换采用什么协议?

地址解析协议 ARP address resolution protocol

39. 请描述 C++的内存管理方式.

在 c++中内存主要分为 5 个存储区：
栈（Stack）：局部变量，函数参数等存储在该区，由编译器自动分配和释放.栈属于计算机系统的数据结构，进栈出栈有相应的计算机指令支持，而且分配专门的寄存器存储栈的地址，效率分高，内存空间是连续的，但栈的内存空间有限。
堆(Heap)：需要程序员手动分配和释放（new,delete），属于动态分配方式。内存空间几乎没有限制，内存空间不连续，因此会产生内存碎片。操作系统有一个记录空间内存的链表，当收到内存申请时遍历链表，找到第一个空间大于申请空间的堆节点，将该节点分配给程序，并将该节点从链表中删除。一般，系统会在该内存空间的首地址处记录本次分配的内存大小，用于 delete 释放该内存空间。
全局/静态存储区:全局变量，静态变量分配到该区，到程序结束时自动释放，包括 DATA 段（全局初始化区）与 BSS 段（全局未初始化段）。其中，初始化的全局变量和静态变量存放在 DATA 段，未初始化的全局变量和静态变量存放在 BSS 段。BSS 段特点：在程序执行前 BSS 段自动清零,所以未初始化的全局变量和静态变量在程序执行前已经成为 0.
文字常量区：存放常量，而且不允许修改。程序结束后由系统释放。

程序代码区：存放程序的二进制代码

40. hash 表的实现，包括 STL 中的哈希桶长度常数。

hash 表的实现主要涉及两个问题：散列函数和碰撞处理。
1）hash function （散列函数）。最常见的散列函数：f(x) = x % TableSize .
2）碰撞问题（不同元素的散列值相同）。解决碰撞问题的方法有许多种，包括线性探测、二次探测、开链等做法。SGL 版本使用开链法，使用一个链表保持相同散列值的元素。
虽然开链法并不要求表格大小必须为质数，但 SGI STL 仍然以质数来设计表格大小，并且将 28 个质数（逐渐呈现大约两倍的关系）计算好，以备随时访问，同时提供一个函数，用来查询在这 28 个质数之中，“最接近某数并大于某数”的质数。

41. hash 表如何 rehash，怎么处理其中保存的资源.

先想想为什么需要 rehash:
因为，当 loadFactor（负载因子）<=1 时，hash 表查找的期望复杂度为 O(1). 因此，每次往 hash 表中添加元素时，我们必须保证是在 loadFactor <1 的情况下，才能够添加。
模仿 C++的 vector 扩容方式，Hash 表中每次发现 loadFactor==1 时，就开辟一个原来桶数组的两倍空间（称为新桶数组），然后把原来的桶数组中元素全部转移过来到新的桶数组中。注意这里转移是需要元素一个个重新哈希到新桶中的。

42. redis 的主从复制怎么做的?

Redis 旧版复制功能只有同步和命令传播。新版复制功能加入了部分同步的功能。
1）同步：
2）命令传播：
当主服务器会将自己执行的写命令，也即是造成主从服务器不一致的那条写命令，发送给从服务器执行，当从服务器执行了相同的写命令之后，主从服务器将再次回到一致状态。
3）部分同步：（断线后重复制）
复制偏移量：通过对比主从服务器的复制偏移量，程序可以很容易地知道主从服务器是否处于一致状态。
复制积压缓冲区：主服务保存最近的写命令到复制积压缓冲区，是一个先进先出队列
服务器运行 ID：从服务器记录上次同步的主服务器的 Id。

43. ubuntu 开机的时候系统做了什么?

1) 加载 BIOS
BIOS 程序首先检查，计算机硬件能否满足运行的基本条件，这叫做”硬件自检”。硬件自检完成后，BIOS 把控制权转交给下一阶段的启动程序。
2) 读取 MBR
计算机读取该设备的第一个扇区，也就是读取最前面的 512 个字节。如果这 512 个字节的最后两个字节是 0x55 和 0xAA，表明这个设备可以用于启动；如果不是，表明设备不能用于启动，控制权于是被转交给”启动顺序”中的下一个设备。
3) Bootloader
在这种情况下，计算机读取”主引导记录”前面 446 字节的机器码之后，不再把控制权转交给某一个分区，而是运行事先安装的”启动管理器”（boot loader），由用户选择启动哪一个操作系统。
Boot Loader 就是在操作系统内核运行之前运行的一段小程序。通过这段小程序，我们可以初始化硬件设备、建立内存空间的映射图，从而将系统的软硬件环境带到一个合适的状态，以便为最终调用操作系统内核做好一切准备。
Boot Loader 有若干种，其中 Grub、Lilo 和 spfdisk 是常见的 Loader。Linux 环境中，目前最流行的启动管理器是 Grub。
4) 加载内核
内核的加载，内核加载后，接开始操作系统初始化，根据进程的优先级启动进程。

44. 程序什么时候应该使用线程，什么时候单线程效率高。

1 耗时的操作使用线程，提高应用程序响应
2 并行操作时使用线程，如 C/S 架构的服务器端并发线程响应用户的请求。
3 多 CPU 系统中，使用线程提高 CPU 利用率
4 改善程序结构。一个既长又复杂的进程可以考虑分为多个线程，成为几个独立或半独立的运行部分，这样的程序会利于理解和修改。
其他情况都使用单线程。

45. 介绍一下模板和容器。如何实现?(也许会让你当场举例实现)

模板可以说比较古老了，但是当前的泛型编程实质上就是模板编程。 它体现了一种通用和泛化的思想。 STL 有 7 种主要容器：
vector,list,deque,map,multimap,set,multiset.

46. C 语言同意一些令人震惊的结构,下面的结构是合法的吗，如果是它做些什么?

inta=5,b=7,c;c=a+++b;
这个问题将做为这个测验的一个愉快的结尾。不管你相不相信，上面的例子是完全合乎语法的。问题是编译器如何处理它?水平不高的编译作者实际上会争论这个问题，根据最处理原则，编译器应当能处理尽可能所有合法的用法。因此，上面的代码被处理成：c = a++ + b; 因此, 这段代码持行后 a = 6, b = 7, c = 12。 如果你知道答案，或猜出正确答案，做得好。如果你不知道答案，我也不把这个当作问题。我发现这个问题的最大好处是:这是一个关于代码编写风格，代码的可读性，代码的可修改性的好的话题

47. #include 与#include “file.h”的区别?

前者是从 Standard Library 的路径寻找和引用 file.h，而后者是从当前工作路径搜寻并引用 file.h。

48. 内存的分配方式有几种?

1)从静态存储区域分配。内存在程序编译的时候就已经分配好，这块内存在程序的整个运行期间都存在。例如全局变量。
2)在栈上创建。在执行函数时，函数内局部变量的存储单元都可以在栈上创建，函数执行结束时这些存储单元自动被释放。栈内存分配运算内置于处理器的指令集中，效率很高，但是分配的内存容量有限。
3)从堆上分配，亦称动态内存分配。程序在运行的时候用 malloc 或 new 申请任意多少的内存，程序员自己负责在何时用 free 或 delete 释放内存。动态内存的生存期由我们决定，使用非常灵活，但问题也最多。

49. 如何让局部变量具有全局生命期。

具体的生命期的概念我觉得我还要好好深入的学习一下，但是这个题目还算比较简单，即用 static 修饰就可以了，但是只是生命期延长，范围并没有扩大，除非把这个变量定义在函数体外的静态区，不过那样就变成全局变量了，仿佛不符合题目要求。

50. strtok 函数在使用上要注意什么问题。

这个问题我不知道能不能回答全面，因为实在是用的很少。这个函数的作用是分割字符串，但是要分割的字符串不能是常量，这是要注意的。比如先定义一个字符串：char array[]=”part1,part2″;，strtok 的原形是 char *strtok(char *string, char *delim);，我们将”,”作为分隔符，先用 pt=strtok(array,”,”);，得到的结果 print 出来就是”part1″，那后面的呢，要写成 pt=strtok(NULL,”,”);，注意，要用 NULL，如果被分割的字符串会被分成 N 段，那从第二次开始就一直要用 NULL。总结起来，需要注意的是：被分割的字符串和分隔符都要使用变量;除第一次使用指向字符串的指针外，之后的都要使用 NULL;注意使用这个函数的时候千万别把指针跟丢了，不然就全乱了。

51. 用预处理指令#define 声明一个常数，用以表明 1 年中有多少秒(忽略闰年问题)

#define SECONDS_PER_YEAR (60 * 60 * 24 * 365)UL 我在这想看到几件事情：
1). #define 语法的基本知识(例如：不能以分号结束，括号的使用，等等)
2). 懂得预处理器将为你计算常数表达式的值，因此，直接写出你是如何计算一年中有多少秒而不是计算出实际的值，是更清晰而没有代价的。
3). 意识到这个表达式将使一个 16 位机的整型数溢出-因此要用到长整型符号 L,告诉编译器这个常数是的长整型数。
4). 如果你在你的表达式中用到 UL(表示无符号长整型)，那么你有了一个好的起点。记住，第一印象很重要。

52. 有 A、B、C、D 四个人，要在夜里过一座桥。他们通过这座桥分别需要耗时 1、2、5、10 分钟，只有一支手电，并且同时最多只能两个人一起过桥。请问如何安排，能够在 17 分钟内这四个人都过桥?

A & B -->2 mins
1 mins <-- A
C & D -->10 mins
2 mins <-- B
A & B --> 2 mins
一共 2 + 1 + 10 + 2 + 2 = 17 mins

53. 1-20 的两个数把和告诉 A,积告诉 B，A 说不知道是多少，B 也说不知道，这时 A 说我知道了，B 接着说我也知道了，问这两个数是多少?

2 和 3

54. 从 300 万字符串中找到最热门的 10 条搜索的输入信息是一个字符串，统计 300 万输入信息中的最热门的前 10 条，我们每次输入的一个字符串为不超过 255byte，内存使用只有 1G。请描述思想，写出算法（c 语言），空间和时间复杂度。

300 万个字符串最多（假设没有重复，都是最大长度）占用内存 3M*1K/4=0.75G。所以可以将所有字符串都存放在内存中进行处理。

可以使用 key 为字符串（事实上是字符串的 hash 值），值为字符串出现次数的 hash 来统计每个每个字符串出现的次数。并用一个长度为 10 的数组/链表来存储目前出现次数最多的 10 个字符串。

这样空间和时间的复杂度都是 O(n)。

55. 如何找出字典中的兄弟单词。给定一个单词 a，如果通过交换单词中字母的顺序可以得到另外的单词 b，那么定义 b 是 a 的兄弟单词。现在给定一个字典，用户输入一个单词，如何根据字典找出这个单词有多少个兄弟单词？

使用 hash_map 和链表。

首先定义一个 key，使得兄弟单词有相同的 key，不是兄弟的单词有不同的 key。例如，将单词按字母从小到大重新排序后作为其 key,比如 bad 的 key 为 abd,good 的 key 为 dgoo。

使用链表将所有兄弟单词串在一起，hash_map 的 key 为单词的 key，value 为链表的起始地址。

开始时，先遍历字典，将每个单词都按照 key 加入到对应的链表当中。当需要找兄弟单词时，只需求取这个单词的 key，然后到 hash_map 中找到对应的链表即可。

这样创建 hash_map 时时间复杂度为 O(n)，查找兄弟单词时时间复杂度是 O(1)。

56. 找出数组中出现次数超过一半的数，现在有一个数组，已知一个数出现的次数超过了一半，请用

O(n)的复杂度的算法找出这个数。

答案 1：

创建一个 hash_map，key 为数组中的数，value 为此数出现的次数。遍历一遍数组，用 hash_map 统计每个数出现的次数，并用两个值存储目前出现次数最多的数和对应出现的次数。

这样可以做到 O(n)的时间复杂度和 O(n)的空间复杂度，满足题目的要求。

但是没有利用“一个数出现的次数超过了一半”这个特点。也许算法还有提高的空间。

答案 2：

使用两个变量 A 和 B，其中 A 存储某个数组中的数，B 用来计数。开始时将 B 初始化为 0。

遍历数组，如果 B=0，则令 A 等于当前数，令 B 等于 1；如果当前数与 A 相同，则 B=B+1；如果当前数与 A 不同，则令 B=B-1。遍历结束时，A 中的数就是要找的数。

这个算法的时间复杂度是 O(n)，空间复杂度为 O(1)。

57. n 个空间（其中 n<1M），存放 a 到 a+n-1 的数，位置随机且数字不重复，a 为正且未知。现在第一个空间的数被误设置为-1。已经知道被修改的数不是最小的。请找出被修改的数字是多少。

算法的时间复杂度是 O(

57. 找出被修改过的数字 n 个空间（其中 n<1M），存放 a 到 a+n-1 的数，位置随机且数字不重复，a 为正且未知。现在第一个空间的数被误设置为-1。已经知道被修改的数不是最小的。请找出被修改的数字是多少。

例如：n=6，a=2，原始的串为 5,3,7,6,2,4。现在被别人修改为-1,3,7,6,2,4。现在希望找到 5。

由于修改的数不是最小的，所以遍历第二个空间到最后一个空间可以得到 a 的值。

a 到 a+n-1 这 n 个数的和是 total=na+(n-1)n/2。

将第二个至最后一个空间的数累加获得 sub_total。

那么被修改的数就是 total-sub_total。

58. 设计 DNS 服务器中 cache 的数据结构。

要求设计一个 DNS 的 Cache 结构，要求能够满足每秒 5000 以上的查询，满足 IP 数据的快速插入，查询的速度要快。（题目还给出了一系列的数据，比如：站点数总共为 5000 万，IP 地址有 1000 万，等等）
DNS 服务器实现域名到 IP 地址的转换。
每个域名的平均长度为 25 个字节（估计值），每个 IP 为 4 个字节，所以 Cache 的每个条目需要大概 30 个字节。
总共 50M 个条目，所以需要 1.5G 个字节的空间。可以放置在内存中。（考虑到每秒 5000 次操作的限制，也只能放在内存中。）
可以考虑的数据结构包括 hash_map，字典树，红黑树等等。

59. 序列 seq=[a,b,…z,aa,ab…az,ba,bb,…bz,…,za,zb,…zz,aaa,…]类似与 excel 的排列，任意给出一个字符串 s=[a-z]+(由 a-z 字符组成的任意长度字符串），请问 s 是序列 seq 的第几个。

注意到每满 26 个就会向前进一位，类似一个 26 进制的问题。
比如 ab，则位置为 26*1+2；
比如 za，则位置为 26*26+1；
比如 abc，则位置为 26*26*1+26*2+3；

60. 找出第 k 大的数字所在的位置。写一段程序，找出数组中第 k 大小的数，输出数所在的位置。例如{2，4，3，4，7}中，第一大的数是 7，位置在 4。第二大、第三大的数都是 4，位置在 1、3 随便输出哪一个均可。

先找到第 k 大的数字，然后再遍历一遍数组找到它的位置。所以题目的难点在于如何最高效的找到第 k 大的数。

我们可以通过快速排序，堆排序等高效的排序算法对数组进行排序，然后找到第 k 大的数字。这样总体复杂度为 O(NlogN)。

我们还可以通过二分的思想，找到第 k 大的数字，而不必对整个数组排序。从数组中随机选一个数 t，通过让这个数和其它数比较，我们可以将整个数组分成了两部分并且满足，{x, xx, . . . , t}<{y, yy, . . . }。

在将数组分成两个数组的过程中，我们还可以记录每个子数组的大小。这样我们就可以确定第 k 大的数字在哪个子数组中。

然后我们继续对包含第 k 大数字的子数组进行同样的划分，直到找到第 k 大的数字为止。

平均来说，由于每次划分都会使子数组缩小到原来 1/2，所以整个过程的复杂度为 O(N)。