一：给了一个用递归实现的快排的代码,要求改写成用栈实现的

```
/* 快速排序的非递归实现 */
int partition(int a[], int left, int right)
{
	if(a == NULL)
		return -1;
	if(left >= right)
		return -1;

	int X = a[left];
	int i = left;
	int j = right;

	while(i < j)
	{
		while(i < j && a[j] >= X)
			j--;
		if(i < j)
			a[i++] = a[j];
		while(i < j && a[i] < X)
			i++;
		if(i < j)
			a[j--] = a[i];
	}
	a[i] = X;
	return i;
}

void quick_sort2(int a[], int left, int right)
{
	stack<int> s;
	int mid = partition(a, left, right);

	if(mid == -1)
		return;
	if(left < mid-1)
	{
		s.push(left);
		s.push(mid-1);
	}
	if(mid+1 < right)
	{
		s.push(mid+1);
```

```
            s.push(right);
        }

        while(!s.empty)
        {
            int r = s.top();
            s.pop()
            int l = s.top();
            s.pop();

            mid = partition(a, l, r);

            if(mid == -1)
                return;
            if(l < mid-1)
            {
                s.push(l);
                s.push(mid-1);
            }
            if(mid+1 < r)
            {
                s.push(mid+1);
                s.push(r);
            }
        }
}
```

二:游戏中让玩家参与抽奖,抽装备.玩家先被等概率传送到十二个房间(对应十二星座),第 i 个房间中拿到装备的概率是 i/50. 玩家抽奖失败后可以花 100 金币再抽一次(第一次不用),如果抽中了则不能再抽. 先是问要抽到装备平均要花多少金币; 又问:玩家不喜欢传到 12 个不同房间的设定,现在要求只能传到一个房间,这个房间提供有 12 种装备,要求每种装备被抽中的概率和之前的一样.就此实现一个生成随机数的算法.

三:先是给出了 P,V 原语及信号量的定义,然后有一个场景:一个水果忍者不停的往一个篮子中捡水果,水果有西瓜和梨两种,篮子最多装 10 个水果,装不了就等待.同时鸣人和佐助分别从篮子中拿西瓜和梨吃,只要有的吃就拿,否则就等待.用 PV 原语 写一段伪代码模拟这个过程.

Semaphore empty=10; //篮子里空位的个数

```
Semaphore watermelon=0, pear=0;
Semaphore mutex = 1;
int fruits=0;
ninja()
{
    P(mutex);
    P(empty);
    if(rand() % 2)
        V(watermelon);
    else
        V(pear);
    V(mutex);
}
naruto()
{
    P(mutex);
    P(watermelon);
    V(empty);
    V(mutex);
}
sasuke()
{
    P(mutex);
    P(pear);
    V(empty);
    V(mutex);
}
```

四:给出了跳表的结构,要求实现一个跳表上的查询操作 search(k),然后分析 search 的时间复杂度. 最后再写一个 insert()的操作.

五:有 N 个广告牌(N<=10 万)可以投放广告,有 k 个用户(k<10 亿)在这些广告牌上投放广告.操作 rent(i,j,k)将从 i 到 j 块 广告牌展示用户 k 的广告,如果原来有别的广告就覆盖掉. 操作 query(i)返回第 i 个广告牌上现在投放的是哪个广告. rent 和 query 操作出现的频率相等.要求设计一个数据结构和相应的算法,尽可能快的实现这两种操作.

六:给出了一段英文文献,是关于 code block 的，然后要求根据文献中给出的算法写一段代码. 要用到 STL.

七:判定给定的字符串序列是否是人类基因片段，人类基因片段的特点是：大写字母后边跟相应小写字母，或者是小的基因片段连在一起。写函数判断。

1. LRU 算法

最少使用页面置换算法。LRU 算法的提出，是基于这样一个事实：在前面几条指令中使用频繁的页面很可能在后面的几条指令中频繁使用。反过来说，已经很久没有使用的页面很可能在未来较长的一段时间内不会被用到。因此，我们只需要在每次调换时，找到最少使用的那个页面调出内存。这就是 LRU 算法的全部内容。

2，在 c++中调用 c 函数

```
extern "C" {
        //函数声明
}
```

3，用 typedef 声明函数指针

```
typedef <返回类型> (*<函数类型名>)(参数表)
typedef void (*PF)(int x);
void func1(int x);

PF pFunc;           //声明一个函数指针只需要用 PF 类型名
pFunc = func1;      //此处也可以使用 pFunc = &func1;
```

4，程序内存分配的方式

内存分配方式有三种：

[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， static 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程序员决定 ，使用非常灵活，但如果在堆上分配了空间，就有责任回收它，否则运行的程序会出现内存泄漏，频繁地分配和释放不同大小的堆空间将会产生堆内碎块。

1. 变量在内存地址的分布为：堆-栈-代码区-全局静态-常量数据
2. 同一区域的各变量按声明的顺序在内存的中依次由低到高分配空间（只有未赋值的全局变量是个例外）。
3. 全局变量和静态变量如果不赋值，默认为 0。 栈中的变量如果不赋值，则是一个随机的数据。
4. 编译器会认为全局变量和静态变量是等同的，已初始化的全局变量和静态变量分配在一起，未初始化的全局变量和静态变量分配在另一起。
5. 主函数中栈的地址都要高于子函数中参数及栈地址，证明了栈的伸展方向是由高地址向低地址扩展的。

5，进程和线程的内存共享，堆和栈哪个公有哪个私有

代码区、全局数据区、堆区是各线程公有的，栈区是各线程独立的。

6，库函数调用是否跟操作系统有关

无关

7，内核态和什么态，系统调用时的上下文切换

8，tcp/ip 协议中哪些协议属于网络层

IP，ICMP，OSPF，EIGRP，IGMP，RIP，ARP，RARP 等

9，给一段程序找 bug

10，进程和线程之间的通信方式

进程间的通信方式：
1.管道（pipe）及有名管道（named pipe）：
管道可用于具有亲缘关系的父子进程间的通信，有名管道除了具有管道所具有的功能外，它还允许无亲缘关系进程间的通信。
2.信号（signal）：
信号是在软件层次上对中断机制的一种模拟，它是比较复杂的通信方式，用于通知进程有某事件发生，一个进程收到一个信号与处理器收到一个中断请求效果上可以说是一致的。
3.消息队列（message queue）：
消息队列是消息的链接表，它克服了上两种通信方式中信号量有限的缺点，具有写权限得进程可以按照一定得规则向消息队列中添加新信息；对消息队列有读权限得进程则可以从消息队列中读取信息。
4.共享内存（shared memory）：
可以说这是最有用的进程间通信方式。它使得多个进程可以访问同一块内存空间，不同进程可以及时看到对方进程

中对共享内存中数据得更新。这种方式需要依靠某种同步操作，如互斥锁和信号量等。
5.信号量（semaphore）：
主要作为进程之间及同一种进程的不同线程之间得同步和互斥手段。
6.套接字（socket）；
这是一种更为一般得进程间通信机制，它可用于网络中不同机器之间的进程间通信，应用非常广泛。

线程之间的同步通信：
1.信号量 二进制信号量 互斥信号量 整数型信号量 记录型信号量
2.消息 消息队列 消息邮箱
3.事件 event

11，结构体的 size-内存对齐问题

32 位系统默认是 4 字节对齐，64 位系统默认是 8 字节对齐

12，几道概率题，包括一个电路问题，一个租自行车的分布

13，算一个最小生成树的权重

图的遍历

14，图的存储方式

邻接矩阵、邻接表、十字链表、邻接多重表

15，满 k 叉树第 k 层第一个结点的编号

k^(k-1)

16，对一个顺序数组,冒泡,堆排序,快排,归并,最快和最慢

冒泡 O(n^2)，快排 O(nlogn)，归并 O(nlogn)，堆排序 O(nlogn)

17，构造和析构函数，T *p=new T[n]; delete p; T 的构造函数和析构函数各有一个输出信息,问程序执行后输出什么

18，一个满二叉树,知道前序遍历结果,求后续遍历结果

1、对于一个内存地址是 32 位、内存页是 8KB 的系统。0X0005F123 这个地址的页号与页内偏移分别是多少。

页面大小是 8KB，那么页内偏移量是从 0x0000（0）~ 0x1FFF（2 的 13 次方 - 1）。0x5F123/8K=2E，余数是 1123；则页号是 47 页，页内偏移量应该是 0X00001123。

2、如果 X 大于 0 并小于 65536，用移位法计算 X 乘以 255 的值为：

(X<<8)-X。X<<8-X 是不对的，因为移位运算符的优先级没有减号的优先级高，首先计算 8-X 为 0，X 左移 0 位还是 8。

3、一个包含 n 个节点的四叉树，每个节点都有四个指向孩子节点的指针，这 4n 个指针中有 ____个空指针。

n 个节点的二叉树，一共有 2n 个指针，n-1 个非空指针，故空指针为 2n-(n-1)个
同理 n 个节点的四叉树，一共有 4n 个指针，n-1 个非空指针，故空指针为 4n-(n-1)个

4、以下两个语句的区别是：

```cpp
int *p1 = new int[10];
int *p2 = new int[10]();
```

第一个动态申请的空间里面的值是随机值，第二个进行了初始化，里面的值为 0

5、计算机在内存中存储数据时使用了大、小端模式，请分别写出 A=0X123456 在不同情况下的首字节

大端模式：0X12　　小端模式：0X56
X86 结构的计算机使用 **小端**模式。一般来说，大部分用户的操作系统（如 windows, FreeBsd,Linux）是小端模式的。少部分，如 MAC OS，是大端模式 的。

6、在游戏设计中，经常会根据不同的游戏状态调用不同的函数，我们可以通过函数指针来实现这一功能，请声明一个参数为 int *，返回值为 int 的函数指针：

int (*func)(int *)

7、下面程序运行后的结果为：

```cpp
char str[] = "glad to test something";
    char *p = str;
    p++;
    int *p1 = static_cast<int *>(p);
    p1++;
    p = static_cast<char *>(p1);
    printf("result is %s\n",p);
```

result is to test something

8、在一冒险游戏里，你见到一个宝箱，身上有 N 把钥匙，其中一把可以打开宝箱，假如没有任何提示，随机尝试，问：

(1)恰好第 K 次（1<=K<=N）打开宝箱的概率是多少。

(N-1)/N * (N-2)/(N-1) * (N-3)/(N-2) * … *1/2 = 1/N

(2)平均需要尝试多少次。

1/N * 1 + 1/N * 2 + … + 1/N * N = (N+1)/2

9、头文件中 ifndef / define / endif 是做什么用的？

防止头文件重复包含以及防止变量被重复定义

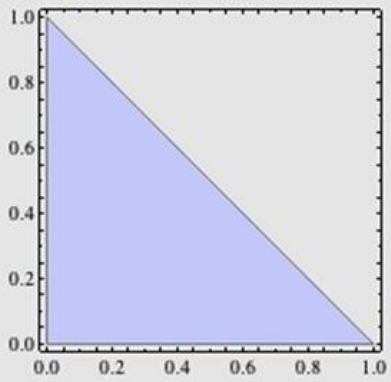
10、代码里有时可以看到 extern “C”，这语句是做什么用的？

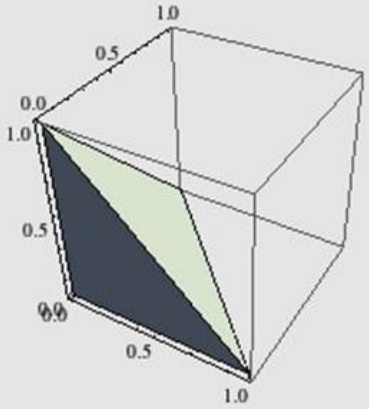
C 和 C++对函数的处理方式是不同的，extern "C"是使 C++能够调用 C 写作的库文件的一个手段，如果要对编译器提示使用 C 的方式来处理函数的话，那么就要使用 extern "C"来说明。

11、平均要取多少个(0，1)中的随机数才能让和超过 1。

为了证明这一点，让我们先来看一个更简单的问题：任取两个 0 到 1 之间的实数，它们的和小于 1 的概率有多大？容易想到，满足 x+y<1 的点 (x, y) 占据了正方形 (0, 1)×(0, 1) 的一半面积，因此这两个实数之和小于 1 的概率就是 1/2 。类似地，三个数之和小于 的概率则是 1/6 ，它是平面 x+y+z=1 在单位立方体中截得的一个三棱锥。这个 1/6 可以利用截面与底面的相似比关系，通过简单的积分求得：

∫(0..1) (x^2)*1/2 dx = 1/6





可以想到，四个 0 到 1 之间的随机数之和小于 1 的概率就等于四维立方体一角的"体积"，它的"底面"是一个体积为 1/6 的三维体，第四维上对其进行积分便可得到其"体积"

∫(0..1) (x^3)*1/6 dx = 1/24

依此类推， n 个随机数之和不超过 1 的概率就是 1/n! ，反过来 n 个数之和大于 1 的概率就是 1 - 1/n! ，因此加到第 n 个数才刚好超过 1 的概率就是

(1 - 1/n!) - (1 - 1/(n-1)!) = (n-1)/n!

因此，要想让和超过 1 ，需要累加的期望次数为

Σ(n=2..∞) n * (n-1)/n! = Σ(n=1..∞) n/n! = e

12、在下列乘法算式中，每个字母代表 0~9 的一个数字，而且不同的字母代表不同的数字：

```
   ABCDEFGH
*          AJ
---------------------
   EJAHFDGKC
   BDFHAJEC
---------------------
   CCCCCCCCC
```

请写出推导的过程。

本题唯一解为：A=2、B=4、C=6、D=9、E=1、F=3、G=5、H=8、J=7、K=0

1、24 小时内，表的时针、分针、秒针完全重合多少次？分别是什么时刻。

设时针的角速度是ω（ω=π/6 每小时），则分针的角速度为 12ω，秒针的角速度为 720ω。

分针与时针再次重合的时间为 t，则有 12ωt-ωt=2π，t=12/11 小时

分针与秒针再次重合的时间为 t，则有 720ωt-12ωt=2π，t=1/59 小时

令 M,N 为正整数，则有(1/59)M=(12/11)N

M=(59*12/11)N

由于 M N 为正整数，所以 N 最小为 11，则 T 至少为

T=（12/11）N=12（小时）

故一天中三针重合只有 0 点和 12 点

2、100 万条记录，可以通过权值比较大小，选取权值最大的前 100 条记录，并有序排列后输出

3、链式数据结构，检测是否有环

4、用反射创建 ClassA 的实例和 ClassA 数组的实例

5、联合索引的问题。问 select 语句如何写，才能用到联合索引

6、cookie 按生命周期分成几类？可以设置哪些属性

7、https 的工作原理和适用范围

8、实现对 Map 对象的 value 进行升序排序

9、多线程实现，ThreadLocal 用法和适用场景

10、ATM 转账流程图和时序图

1.写出对“知之者不如好之者，好之者不如乐之者”的理解。

2.用中文写出尽可能多的中文语句，要求包含有“都”的意思（all、both 之意），但不能有“都”字

3.new/delete 和 malloc/free 的区别，并说说你在什么情况下会自另行建立自己的内存分配机制。

相同点：都可用于申请动态内存和释放内存

不同点：1. malloc 与 free 是 C++/C 语言的标准库函数，new/delete 是 C++的运算符。

2. new 自动计算需要分配的空间，而 malloc 需要手工计算字节数

3. new 是类型安全的，而 malloc 不是，比如：

```
int* p = new float[2]; // 编译时指出错误
int* p = malloc(2*sizeof(float)); // 编译时无法指出错误
```

4. new 将调用 constructor，而 malloc 不能；delete 将调用 destructor，而 free 不能。

在那些需要动态分配大量的但很小的对象的应用程序里，另行建立自己的内存分配机制可以提高效率。

4.求极限 lim（x-[x]),x 趋于-3。

解：x 从-4 趋于-3 时，极限为 1；从 x 从-2 趋于-3 时，极限为 0；

5.比较两个电路的可靠性。

6.编程题：输入一个正整数，若该数能用几个连续正整数之和表示，则输出所有可能的正整数序列。

```
void sum(int num)
{
	int first = 1;
	int end = 2;
	int sum = 0;
	for(int i = first; i <= end; i++)
		sum += i;
	while(first <= num)
	{
		if(sum < num)
		{
			end++;
			sum += end;
		}
		else if(sum > num)
		{
			sum -= first;
			first++;
		}
		else
		{
			for(int i = first; i <= end; i++)
				cout << i << " ";
			cout << endl;
			sum -= first;
```

```
                first++;
            }
        }
```

7.有两个字符串 str1 和 str2，写一个函数实现在 str1 中查找 str2 的初始位置。要求不区分大小写。

A 的 ASCII 码是 65，a 的 ASCII 码是 97

```
char* strstr2(char str1[], char str2[])
{
    if(str1 == NULL || str2 == NULL)
        return NULL;
    char s1[4096], s2[4096];
    int i=0;
    char *p=str1;
    while(p)
    {
        s1[i]=tolower(*p);
        i++;
        p++;
    }
    s1[i]=’\0’;
    i=0;
    p=str2;
    while(p)
    {
        s2[i]=tolower(*p);
        i++;
        p++;
    }
    s2[i]=’\0’;
    return strstr(s1, s2);
}
```

9.求 Fibonacci 数列中第 k 个与前面所有数互质的数（除前面两个数 1，1 ）。

8.在字符串 S 中寻找最长的字符串 x，条件是 x 存在于 S 中。即是如：abcabcdcd 中的 abc。

10.有 100 个真币和一个假币，只知道真币与假币不等重，要求只称两次，得出是真币重还是假币重。

101 个钱币中，任取一个，其余 100 个分为 2 份，如果重量相同，取出的必定假币，与任何一个真币比较即可。如果重量不等，把重的 50 枚分为两分，如果不等，则假币重，否则假币轻。把轻的 50 枚分开称也可以得到结果。

11.证明题：给出 n 个互不相同的分数数列 a1/b1， a2/b2… an/bn ，证明 (a1 + a2 + … + an) / (b1 + b2 + … + bn) 的值在数列 a1/b1， a2/b2… an/bn 数列的最大值和最小值之间。

12. 证明题：在三角形中，假设等角对等边，证明大角对大边。

解：等腰三角形 ABC，其中 b=c

由余弦定理得 cosB = (a^2+c^2-b^2)/2ac = a^2/2ab = a/2b
cosA = (b^2+c^2-a^2)/2ac = (2b^2-a^2)/2b^2 = 1-a^2/2b^2
不妨设 A>B，则 cosA < cosB
1-a^2/2b^2 < a/2b
最后推出 a>b。故大角对大边。

13. 文学题：在以下的空白中填入相应的词（谒、断、淡、泰）并解释其含义。
自处超然，处事 然，无事澄言，有事 言，得意 然，失意 然。
答：自处超然，处事泰然，无事澄言，有事谒言，得意淡然，失意断然。

14. 问答题：为什么现在的计算机采用二进制？而不是八进制或十六进制？你认为以后的计算机会采用几进制？

15. 程序设计题：给出若干个单词，组成字典，要求查找速度最快。

19. 一个没有拷贝构造函数和重载＝运算符的 S tring 类，会出现什么问题，如何解决？
出现问题：浅拷贝；解决方法：按位拷贝

21. 有一位警长，抓了三个逃犯。现警长决定给他们一次机会。他拿出 3 顶黑帽子，两顶白帽子，然后往这三个逃犯头上每人戴了一顶帽子，每个逃犯只能看 到另外两个逃犯帽子的颜色，不能看到自己帽子的颜色，而且不能进行通讯，不能进行讨论，只能靠自己的推理推出来，如果猜出来了，放一条生路，否则处死。
警长先问第一逃犯，结果第一逃犯猜错了，被杀掉了。
警长问第二个逃犯，结果还是猜错了，同样被杀掉了。
警长再问第三个逃犯，结果第三个逃犯猜对了。
说明一下，每个逃犯在回答问题时，其他逃犯是听不到的。
为什么第三个一定能猜中，请你给出解释。
如果 A 看到另外两个人都带白色帽子，那么自己肯定带黑色帽子。
如果 A 看到另外两个帽子一白一黑，而黑色帽子的那个人死了（一白一黑都死了），那么自己肯定不是白帽子，而是黑帽子。
如果 A 看到另外两个帽子都是黑色的，而第二个黑帽子也死了（如果 A 带白帽子，那么地一个黑帽子死了，第二个黑帽子应该知道自己帽子的颜色），那么 A 肯定不是带白帽子，而是带黑帽子。

22. 填词：亲、与、举、不取、不为
居，视其所（ ）
富，视其所（ ）
达，视其所（ ）
穷，视其所（ ）
贫，视其所（ ）
并谈谈你的理解。

居，视其所亲
富，视其所与
达，视其所举

穷，视其所不取
贫，视其所不为
这句话的意思是告诉你怎么观察一个人