**阿里2015实习生-客户端笔试题目解析**

字数4220 阅读5189 评论21 喜欢93

职位：2015实习生-客户端开发工程师  
考试时间：笔试，基本40分钟 + 附加80分钟  
如果你有比我答案更好方法的话，欢迎留言~ 已经修改多次了。

1.在网络传输数据时，经常需要将二进制数据转换为一个可打印字符串，一般用到的可打印字符集合包含64个字符，因此又称为Base64表示法，现有一个char数组长度为12，要将它表示为Base64字符串，请问Base64字符串至少需要几个char；如果char的长度为20，则需要几个char。

解答：  
char是一个基本数据类型。它可以表示一个byte大小的数字，即8位，而Base64使用基于6位的编码。所以就是12x 8/6 = 16;而20 x 8/6 = 20 x 4/3 = 24 + 2.666 ，多余的需要用 ====补齐到4位，所以是28。

答案：16，28

应用实例:

1. 在HTTP GET/POST中一般都需要对参数进行base64编码
2. 在OSX中的.plist文件中的<data>数据也是Base64编码的

2.某部门年会的时候，有个环节叫做《国王和天使》，每个人将自己的名字写在纸条上并丢入盒子中，等所有人都完成该动作后，每个人再从盒子里拿出一张小纸条，上面的人物就是自己的国王，自己就是ta的天使，如果抽到自己，就再抽一遍，直到抽到别人为止，如果用有向图来表示他们之间的关系，那么以下结论错误的是*\_*?

1. 最后的结果可能是存在多个联通分支

2. 联通分支个数最多不超过人数的二分之一

3. 最后的结果可能是存在多个强联通分支

4. 可以用深度优先算法求得联通数

5. 可以用双向链表存储其结构

6. 遍历的复杂度是O(N)

解答：  
这个题目有三个BUG，当是人数三个人（或者奇数）时，其中两个人互相抽取的话，总有一个人抽不到，这个人就活该单身吗？还有个问题就是，在数学中一般翻译为“连通”而不是“联通"; 最后一个问题就是，连通数这个概念我没有在离散的书上找到，谷歌上也很少搜到。所以我认为这是一个不严谨不负责任的题目。

接下来是我的分析，我们先假设，当出现四个人的时候(偶数特例)，两两成环的话，答案123都是对的。

接着答案五也是对的，这个结构明显就是链表。

第六的复杂度等于 O(线 + 边)，肯定在2N左右，所以也是对的。

最后我们排除出来的是4。对于4的理解，我认为它只能求出连通通路的个数，而不是连通数。

所以，答案是4。

是不是看晕了？看来需要自学一遍离散了 ---> [点我](http://www.wlxt.uestc.edu.cn/wlxt/ncourse/lsxx/web/lssx/end/imgs/main.htm)

3.假设支付宝红包口令支持1到6位的数字组合，即'0','1','003'和‘999999’都是合法的红包口令，那么总共可以有多少个合法的红包口令？

解答：高中概率题，注意'1'，'000001'与'001'是不一样的就可以理解了。10 + 100 + 1000 + 10000 + 100000 + 1000000 = 1111110

4.假设某棵二叉查找树的所有键均为1到10的整数，现在我们要查找下面*\_*是不可能是键的检查序列。

1. 10，9，8，7，6，5

2. 2，8，6，3，7，4，5

3. 1，2，9，3，8，7，4，6，5

4. 2，3，10，4，8，5

5. 4，9，8，7，5

6. 以上均正确

解答：  
一个个按照BST画呗，然后找出有分叉的，第二个是错的。

5.能整除1400的数目是\_\_?

解答：

笔试的时候我直接跳了，毕竟2分钟算这个划不来。

后来问了学长，有了一个新的方法。

首先 1400 = (2x2x2) x (5x5x7）

然后我们计算 2x2x2 ，有 2 4 8 三个

计算 5x5x7 , 有 5 7 25 35 175 共5个

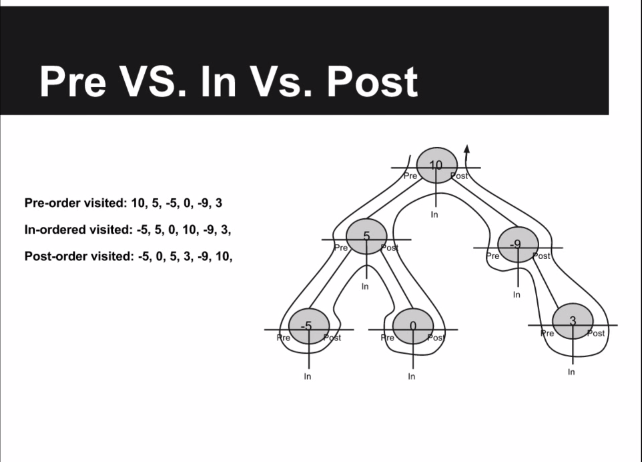
这两个相乘，有 15 个

最后就是 15 + 3 + 5 = 23 个。

（我看到百度知道上有人在考试时间去问题目，非常鄙视这种行为）

6.设二叉树节点的线序遍历，中序遍历与后序遍历，所有叶子节点的先后顺序*\_\_*？

解答：  
三者是相同的，遍历顺序不影响叶子节点



*PreOrder Travesal*

7.一个具有513个节点的二叉树，有*\_*种可能的层高。

解答：  
513 = 511 + 2

* 如果是完全树的话，就是  
  2^count - 1= 511 => count = 9 ,完全树是9层，然后加上一个小尾巴，就是10层。
* 接着，我们考虑最差的情况，就是树退化为链表，这时 count = 513;

所以[10,513]，就是513-10 + 1 = 504;

8.给定一个整数sum，从有N个无序元素的数组中寻找元素a，b，c，d，使得a + b + c + d = sum，最快的平均时间复杂度是*\_*?

1. O(N^2)

2. O(log N)

3. O(N)

4. O(N^3)

5. O(N^2logN)

6. O(N^4)

解答：这个问了学长。

寻找一个元素至少需要N个时间；  
寻找两个元素的话，俩俩相加，消耗N-1时间，然后去遍历。总时间约为2N  
寻找三个元素的话......总时间为3N

反正最后就是4N，也就是O(N)

9.如下SQL数据中,*\_*可能返回null值。

1. select count(\*) from t1;//返回行的个数

2. select max(col1) from t1;//返回列中的某个最大值

3. select concat('max=' ,maxof(col1)) from t1;//连接字符

解答：  
如果concat连接的字符中参数为空的话，则返回为空。  
<http://zhidao.baidu.com/question/1669729226131857827>

10.函数lg(ax^2 - 8x + a - 6)的值域为R，那么实数a的取值范围是*\_*？

解答：

ax^2 - 8x + a - 6 > 0

当 a > 0 ，Δ > 0

64 - 4a^2 + 24a > 0,

16 - a^2 + 6a > 0

a^2 - 6a -16 <0

(a + 2)(a - 8) >0

so , a>8

当 a < 0 , no way.

so , a>8

11.一个电动模型，每一组电池能让其行驶8分钟，一个充电器能同时给两组电池充电，一组充满需要15分钟，至少准备*\_*组电池，可以让模型行驶完立即换电池行驶不用等待。

解答：  
两组肯定不够的，假设有3组充满的电池，我们用笔划一划

int use = 8

int charge[2] = 0,0

use = 0

charge[2] = 0,0;

use = 8

charge = 15,0

use = 0

charge = 7,0;

use = 8

charge = 7,15;

use = 0;

charge = 15, 8;

所以3个是可以的。

12.对于下面的代码，正确的是？

char\* s1 = "Hello world";

char s2[] = "Hello world";

s1[2] = 'E'; //1

s2[2] = 'E'; //2

\*(s1 + 2) = 'E'; //3

\*(s2 + 2) = 'E'; //4

解答：  
s1是 char\*类型，它指向常量字符串，而常量早已经在编译的时候就写入程序中了，是不可改变的;  
s2是 char[]类型，它指向数组的第一位；

我们分开解答，先把情况1转换为单独的代码

void dosome(void){

char\* s1 = "hello world";

s1[2] = 'E';

}

系统报错

Bus error: 10

我们拿出Hopper Disassembler这个神器，把二进制文件反编译后是

function dosome() {

var\_m8 = "hello world";

rax = var\_m8;

//int8\_t 就是 char

\*(int8\_t \*)(rax + 0x2) = 0x45;

return rax;

}

通过反编译，我们知道了s1[2] = 'E' 实际上就是先强制转换，然后所指向的值赋值为 0x45 的意思。但是常量不能修改，所以会报错，而且修改常量也没有任何意义。

而 \* (s1 + 2)同样，反编译的结果与上面相同，所以也是错的。

接下来我们分析数组，我们同样拆分为单独的函数，便于反编译

void dosome(void){

char s2[] = "hello world";

s2[2] = 'E';

}

反编译发现，编译器自动生成了多个匿名整型值。

function dosome() {

rax = \*\_\_stack\_chk\_guard;

var\_24 = \*rax;

var\_12 = \*"hello world";

//这里面实际就是读取\_DATA中的低位端，s[2],也就是'l'。注意看这里并没有使用val\_12

var\_20 = \*(int32\_t \*)0x100000f9a;

var\_14 = 0x45;

if (\*rax == var\_24) {

return \*\_\_stack\_chk\_guard;

}

else {

rax = \_\_stack\_chk\_fail();

}

return rax;

}

这个当然能运行，改成 \*(s +2)仍然能运行，反编译代码也一样。

综上我们有如下结论：

1. char\* 定义的字符串常量不能修改，而且为了安全一般定义为const的，这道题作者有病没有加上;
2. 用数组定义的字符串可以修改;
3. s[2] == *(s + 2) ==*(intX\_t \*)(s + 2);

所以1，3错的。

13.一台刚接入互联网的WEB服务器第一次被访问到时，不同协议的发生顺序是下面中的*\_*？

1.ARP->DNS->HTTP

2.ARP->HTTP->DNS

3.DNS->HTTP->ARP

4.DNS->ARP->HTTP

5.HTTP->ARP->DNS

6.HTTP->DNS->ARP

解答：  
有人问了，结果比我整理的好。  
服务器插上网线需要在内网进行IP-MAC映射，也就是ARP。接着用户在浏览器输入地址，查找DNS，获取IP，最后以TCP的形式把HTTP封装，进行连接。  
<http://www.zhihu.com/question/29282648>

[在浏览器中输入Google.com并且按下回车之后发生了什么？](http://kb.cnblogs.com/page/516964/)

14.忘了 %>\_<%

15将整数序列(7-2-4-6-3-1-5-8)按所示顺序构建一颗二叉排序数a，之后将整数4按照二叉树排序规则从树a中删除，请问删除后树的前序遍历结果是\_？

1. 1-2-3-4-5-6-7

2. 7-2-1-5-3-6-8

3. 3-6-1-5-2-8-7

4. 1-3-6-5-2-8-7

5. 8-7-6-5-3-2-1

6. 7-2-8-1-5-3-6

解答：二叉排序树，也叫二叉搜索树(Binary Search/Sort Tree)，按照left < mid < right进行排序。找出4右树的最小值，也就是5,然后替换即可。前序遍历的话，看刚刚的三个遍历对比图就可以秒杀了，所以选择2

BST动画 ——>[点我](http://www.comp.nus.edu.sg/~stevenha/visualization/bst.html)

16.以下程序运行结果？

#include<iostream>

using namespace std;

class cat{

int age;

public:

//构造函数

cat(){age = 1;}

//析构函数，类似于Java的finalize

~cat(){}

//一个公共方法

void miao(void){

cout<<"Miao,my age = "<<age<<endl;

}

};

int main(void){

cat kitty;

int\* p = (int\*)(&kitty);

\*p = 20;

kitty.miao();

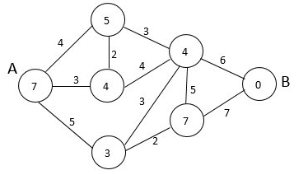
return 0;

}

解答：  
实际输出的是Miao,my age = 20;类的内存分配类似于结构体，它的内部就是一个int，\*p = 20实际上就是

mov dword[ss:rbp + p],0x14;

17.图中的每个圈圈的一个补给站，存储着一定数量的汽油，每个圈之间的路上标识来了这段路需要消耗的汽油量，一段小车从A点出发，在图上任意行走，到达补给站的时候可以取出所有的汽油，到达B点后最多剩余的汽油量是\_\_?



*阿里面试*

5,6,7,9,10,13中的哪一个？

解答：  
这个是加权图，深度遍历，必须要纸笔画出来。花时间计算有些划不来，答案是10。如果猜的话，一般是倒数第二个 😊。

我的路线是7-3+4-2+5-3+4-3+3-2+7-7 = 10

18.以下关于STL的描述中，*\_*是错的。

1. STL容器是线程不安全的

2. 当容量不够时，vector内部内存扩展方式是翻倍

3. std::sort是稳定排序

4. std::string中可以存储多个'\0'字符

5. std::bitset不是一个STL容器

6. std::stack默认是用deque实现的

解答：

* STL类似于JDK里面的“伸手库”,帮你封装好了很多常见的结构与算法
* STL是部分线程安全的。
* sdt::sort 中间的冒号" :: "是作用域限定符(field qualifier)或称作用域运算符，用它声明函数是属于哪个类的，类似于JAVA中的Arrays.sort()；
* 剩下的问题都跟Java类似，我是来面试JAVA的！！！

所以选1

19.忘记了，好像是数组指针混在一起的题目，

20.求自增与优先级的，具体代码不清楚了，反正就是这个代码

int calculate(int y,int t){

return ++y | t++;

}

题目到底是几我们不关心，我们只需要知道

* 自增的优先级高于二进制运算，所以要全部自增运算完，才可以或运算
* 关于++var与var++的问题， 我们常见的i++都是不影响结果的，而++i才影响结果

反编译后的代码是

function calculate(int, int) {

var\_m4 = rdi;

var\_m8 = rsi;

rsi = var\_m4 + 0x1;

var\_m4 = rsi;

rdi = var\_m8;

var\_m8 = rdi + 0x1;

return rsi + rdi;

//更简单的说，就是y++; x=y|t; t++;

}

也就是说，比如我们计算calculate(1,9)就是计算

1 + 1;

2 | 9;

return 2 | 9;

9 + 1;

这几个步骤.

附加题：

这部分为自由题目，，反复改了好多次，发现这个题目不是简单的“阻塞”能解决的。

1.请列举一些导致"APP启动慢"感受的典型问题或负面模式及相应的优化策略。

以Android支付婊为例：

1. UI阻塞问题

比如布局深度，过度渲染，内存抖动，耗时任务，导致UI阻塞，但是这个我觉得连我这样的菜鸟都知道的优化，人家早就解决了啊（这个如果是个人开发的话，可以用GPU profile进行UI调试，移除阻塞代码，有一年经验的人都会的）。

2. 业务太多

假设支付宝使用了异步任务解决了阻塞问题，为什么仍然卡呢？我认为是支付宝的业务太多（并发过多，框架加载过多）。比如

1. 启动时连接各种不同域名的HTTPS比较耗时，多线程也撑不住啦
2. 启动时候就开始互相唤醒各种毒瘤进程，类似于病毒的双守护线程（比如淘宝）；
3. 启动各种后台Service；
4. 启动一大堆单例框架，供全局调用；
5. 加载各种JNI组件
6. 由于支付宝对安全要求高，需要更多的加解密运算，看看data下的so文件大家就知道了；

解决办法：

1. 简化HTTPS的API获取流程，以减少并发连接量；
2. 在加载各种单例或者框架时，进行加载调度（说的容易）
3. 使用JNI解决密集运算（人家的确这么做了）

3. 团队协作

可以说，支付宝的业务可能比一个JAVA EE服务器还要多，这时需要一个好的架构师和一个好的团队。

1. 架构师当初对支付宝框架搭建不合理，而且现在历史旧帐太多，非常不好修改了；
2. 团队是从服务器平台调过来的，可能不太适应嵌入式下资源紧缺的环境（这个是我脑补的，我上次看到一个JAVAEE码农写的Android代码，唉~）；

4. 更多的优化

可以看看[如何从程序优化的角度解释淘宝支付宝的安卓版卡顿？](http://www.zhihu.com/question/30033704/answer/46641046?utm_source=weibo&utm_medium=weibo_share&utm_content=share_answer&utm_campaign=share_button)

2.客户端开发不同于服务端开发的一个典型背景是，我们的程序会运行在大量不同的设备商，带给我们很多底层执行环境上的约束，例如Android生态中典型的碎片化问题，有的团队为此只适配主流机型，有的约束开发可以使用的API，有的干脆只为iOS平台开发。请在平衡用户体验与研发成本之上，站在App设计和研发的角度，提出你的对策。

这道题目又把1%的用户给忘了，好黑.....

我的看法是：  
对于Android来说：

1. 适配4.0以上的设备；
2. Flyme与MIUI为了所谓的个性化与安全，需要特别进行适配，比如短信功能。
3. 为了缩减成本，尽量使用优秀的通用的框架，同时贡献代码；

3.请以你的平台为例，解释APP“卡”的原因，并给出“滑屏卡顿”与“转场卡顿”的解决方案。

这道题目还是优化问题，同上。

在Android中，我们在开发者选项中打开GPU profile，然后在AndroidStudio对内存进行监控。