设图 G 的相邻矩阵如下图:则 G 的顶点数和边数分别为()

```
01111
10100
11011
10101
10110
```

正确答案: D 你的答案: 空 (错误)

```
4,5
4,10
5,6
5,8
```

mysql 数据库有选课表 learn(student_id int,course_id int),字段分别表示学号和课程编号, 现在想获取每个学生所选课程的个数信息,请问如下的 sql 语句正确的是()

正确答案: B 你的答案: 空 (错误)

```
select student_id,sum(course_id)from learn
select student_id,count(course_id)from learn group by student_id
select student_id,count(course_id)from learn
select student_id,sum(course_id)from learn group by student_id
```

下列排序算法中元素的移动次数和关键字的初始排列次序无关的是()

正确答案: C 你的答案: 空 (错误)

直接插入排序
起泡排序
基数排序
快速排序

某一密码仅使用 K、L、M、N、O 共 5 个字母,密码中的单词从左向右排列,密码单词 必须遵循如下规则:
(1) 密码单词的最小长度是两个字母,可以相同,也可以不同
(2) K 不可能是单词的第一个字母
(3) 如果 L 出现,则出现次数不止一次
(4) M 不能使最后一个也不能是倒数第二个字母 (5) K 出现,则 N 就一定出现
(6) O 如果是最后一个字母,则 L 一定出现 问题:下列哪一个是单词密码?()

正确答案: C 你的答案: 空 (错误)

```
KLLN
LOML
MLLO
```

NMKO

n 从 1 开始,每个操作可以选择对 n 加 1,或者对 n 加倍。如果最后结果为 2013,最少 需要()个操作。

正确答案: A 你的答案: 空 (错误)

18
24
21
不可能

下面一段代码的输出结果是()

```
1    void f(char *c,  char d) {
2            *c = *c + 1;
3            d = d + 1;
4            cout << *c << d;
5    }
6    int main() {
7            char a =  'A', b =  'a'; f(&b, a);
8            cout << a << b << endl;
9            return 0;
10   }
```

正确答案: B 你的答案: 空 (错误)

BaBa
bBAb
AbAb
aBaB

关于 JAVA 堆,下面说法错误的是()

正确答案: D 你的答案: 空 (错误)

所有类的实例和数组都是在堆上分配内存的
对象所占的堆内存是由自动内存管理系统回收
堆内存由存活和死亡的对象,空闲碎片区组成
数组是分配在栈中的

某系统有 n 台互斥使用的同类设备,3 个并发进程需要 3,4,5 台设备,可确保系统不发生死锁的设备数 n 最小为()

正确答案: B 你的答案: 空 (错误)

9
10
11

12

一堆硬币,一个机器人,如果是反的就翻正,如果是正的就抛掷一次,无穷多次后,求 正反的比例()
正确答案: B 你的答案: 空 (错误)

3:1
2:1
4:1
6:1

主机甲和乙已建立了 TCP 连接,甲始终以 MSS=1KB 大小的段发送数据,并一直有数据 发送;乙每收到一个数据段都会发出一个接收窗口为 10KB 的确认段。若甲在 t 时刻发生超 时时拥塞窗口为 8KB,则从 t 时刻起,不再发生超时的情况下,经过 10 个 RTT 后,甲的发送窗口是()
正确答案: A 你的答案: 空 (错误)

10KB
12KB
14KB
15KB

下面哪几种是 STL 容器类型()
正确答案: A B D E 你的答案: 空 (错误)

vector
set
multivector
multiset
array

下面有关 JAVA 异常类的描述,说法正确的有()
正确答案: A C 你的答案: 空 (错误)

异常的继承结构:基类为 Throwable,Error 和 Exception 继承 Throwable,RuntimeException 和 IOException 等继承 Exception
非 RuntimeException 一般是外部错误,其必须被 try{}catch 语句块所捕获
Error 类体系描述了 Java 运行系统中的内部错误以及资源耗尽的情形,Error 不需要捕捉
RuntimeException 体系包括错误的类型转换、数组越界访问和试图访问空指针等等,必须 被 try{}catch 语句块所捕获

下面有关 java 类加载器,说法正确的是?()

正确答案: A B C D　你的答案: 空 (错误)

引导类加载器(bootstrap class loader):它用来加载 Java 的核心库,是用原生代码来实现的
扩展类加载器(extensions class loader):它用来加载 Java 的扩展库。
系统类加载器(system class loader):它根据 Java 应用的类路径(CLASSPATH)来加载 Java 类
tomcat 为每个 App 创建一个 Loader,里面保存着此 WebApp 的 ClassLoader。需要加载 WebApp 下的类时,就取出 ClassLoader 来使用

下面有关事务隔离级别说法正确的是?()
正确答案: A B C D　你的答案: 空 (错误)

串行读(Serializable):完全串行化的读,每次读都需要获得表级共享锁,读写相互都会阻塞
未提交读(Read Uncommitted):允许脏读,也就是可能读取到其他会话中未提交事务修改 的数据
提交读(Read Committed):只能读取到已经提交的数据
可重复读(Repeated Read):在同一个事务内的查询都是事务开始时刻一致的

下列说法错误的是 ()
正确答案: B D　你的答案: 空 (错误)

利用一组地址连续的存储单元依次存放自栈底到栈顶的数据元素,这种形式的栈也称为顺序栈
top=0 时为空栈,元素进栈时指针 top 不断地减 1
当 top 等于数组的最大下标值时则栈满
栈不能对输入序列部分或全局起求逆作用

找出两个链表相交的结点(定义链表结构)

```
structNode
{
  int data;
  structNode *next;
}
class solution
{
  public:
```

```
structNode* FindCommen(structNode* list1,structNode* list2)
{
  if(list1==NULL || list2==NULL)
      return NULL;
 int len1=list1.size();
 int len2=list2.size();
 int diff;
if(len1>len2)
   {
     diff=len1-len2;
     for(int i=0;i<diff;i++)
       {
          list1=list1->next;
       }
   }
else
   {
      diff=len2-len1;
      for(int i=0;i<diff;i++ )
     {
        list2=list2->next;
     }
   }
while(list1->data != list2->data)
{
   list1=list->next;
  list2=list2->next;
}
return list1;
```

```
}
```

给定一个已经排好序的字符串数组,空字符串散布在该数组中,编写一个函数寻找一个 给定字符串的位置。

```
1   //适合懂得指针操作的人看
2   #include<stdio.h>
3   #include<string.h>
1   int findIndex(char* par_str,  char* child_str )
2   {
3     short i,j=0;
4     char* temp = child_str;
5
6     for(i=0; i<strlen(par_str); i++)
7     {
8
9       if(*temp == *(par_str+i))
10      {
11        temp++;
12
13        if( ++j ==  strlen(child_str))
14          return i-j+1;
15      }
16      else
17        temp = child_str;
18    }
19    return -1;
20  }
1    int main(void)
2    {
3      char par_arr[] =  "abc 123 cxy ppppp";
4      char child_str[] =  "123";
5      int index;
6
7      index = findIndex(par_arr, child_str);
8
9      printf("index= %d", index);
10
11     return 0;
12   }
```

给定一个二叉树,且每个节点存储一个值。设计一个算法,实现:对于一个给定的数值,打印出所有的路径。这条路径不必要开始于或结束语根节点或叶节点。

```
bool visitTreePathByValue(TreeNode * pNode , int nNodeVal)
{
if (NULL == pNode)
{
return false;
}
if (nNodeVal == pNode->value())
{
printf("%d" , pNode->value());
return true;
}
bool bVisit = false;
bVisit = visitTreePathByValue(pNode->left() , nNodeVal);
if (bVisit)
{
printf("%d" , pNode->value());
}
bVisit = visitTreePathByValue(pNode->right() , nNodeVal);
if (bVisit)
{
printf("%d" , pNode->value());
}
return bVisit;
}
```