以下哪个协议属于传输层协议？
正确答案: B　你的答案: 空 (错误)

FTP
UDP
IP
HTTP

以下哪个算法不是对称加密算法()
正确答案: D　你的答案: 空 (错误)

DES
RC5
AES
RSA

在多线程系统中，线程在它的生命周期中会处于不同的状态，其中不是线程状态的是:()
正确答案: B　你的答案: 空 (错误)

Ready
Busied
Blocked
Running

设数组 data[m]作为循环队列的存储空间。front 为队头指针，rear 为队尾指针，则执行出队操作后其头指针 front 值为()
正确答案: D　你的答案: 空 (错误)

front=front+1
front=(front+1)%(m-1)
front=(front-1)%m
front=(front+1)%m

下列关于管道(Pipe)通信的叙述中，正确的是()
正确答案: C　你的答案: 空 (错误)

一个管道可以实现双向数据传输
管道的容量仅受磁盘容量大小限制
进程对管道进行读操作和写操作都可能被阻塞
一个管道只能有一个读进程或一个写进程对其操作

已知数组元素基本有序的情况下，下面采用那个算法对数组排序时间复杂度最低()
正确答案: D　你的答案: 空 (错误)

直接选择排序
堆排序
快速排序
插入排序

下面关于 B 和 B+树的描述中，不正确的是()
正确答案: C　你的答案: 空 (错误)

B 树和 B+树都是平衡的多叉树
B 树和 B+树都可用于文件的索引结构
B 树和 B+树都能有效的支持顺序检索
B 树和 B+树都能有效的支持随机检索

关于依赖注入，下列选项中说法错误的是()
正确答案: B　你的答案: 空 (错误)

依赖注入能够独立开发各组件，然后根据组件间关系进行组装
依赖注入使组件之间相互依赖，相互制约
依赖注入提供使用接口编程
依赖注入指对象在使用时动态注入

下列哪个地址不可能是子网掩码()
正确答案: D　你的答案: 空 (错误)

255.224.0.0
255.255.240.0
255.255.255.248
255.255.255.250

若一颗二叉树具有 10 个度为 2 的节点，5 个度为 1 的节点，度为 0 的节点个数为()
正确答案: B　你的答案: 空 (错误)

9
11
15
不确定

在 Java 中，以下关于方法重载和方法重写描述正确的是？
正确答案: D　你的答案: 空 (错误)

方法重载和方法的重写实现的功能相同
方法重载出现在父子关系中，方法重写是在同一类中
方法重载的返回值类型必须一致，参数项必须不同
方法重写的返回值类型必须相同或相容。（或是其子类）

下面有关 JVM 内存，说法错误的是？
正确答案: C　你的答案: 空 (错误)

程序计数器是一个比较小的内存区域，用于指示当前线程所执行的字节码执行到了第几行，是线程隔离的
Java 方法执行内存模型，用于存储局部变量，操作数栈，动态链接，方法出口等信息，是线程隔离的
方法区用于存储 JVM 加载的类信息、常量、静态变量、即使编译器编译后的代码等数据，是线程隔离的
原则上讲，所有的对象都在堆区上分配内存，是线程之间共享的

C++中，下面四个表达式中错误的一项是()
正确答案: C　你的答案: 空 (错误)

```
a+=(a++)
a+=(++a)
(a++)+=a
(++a)+=(a++)
```

多线程中栈与堆是公有的还是私有的()
正确答案: C　你的答案: 空 (错误)

栈公有，堆私有
栈公有，堆公有
栈私有，堆公有
栈私有，堆私有

有如下 4 条语句：()
```
1    Integer i01=59;
2    int i02=59;
3    Integer i03=Integer.valueOf(59);
4    Integer i04=new Integer(59);
```
以下输出结果为 false 的是:
正确答案: C　你的答案: 空 (错误)

```
System.out.println(i01==i02);
System.out.println(i01==i03);
System.out.println(i03==i04);
System.out.println(i02==i04);
```

设有 5000 个待排序的记录的关键字，如果需要用最快的方法选出其中最小的 10 个记录关键字，则用下列哪个方法可以达到此目的()
正确答案: B　你的答案: 空 (错误)

快速排序
堆排序
归并排序
插入排序

三个程序 a,b,c,它们使用同一个设备进行 I/O 操作，并按 a,b,c 的优先级执行(a 优先级最高，c 最低).这三个程序的计算和 I/O 时间如下图所示。假设调度的时间可忽略。则在单道程序环境和多道程序环境下(假设内存中可同时装入这三个程序，系统采用不可抢占的调度策略).运行总时间分别为()

	计算	I/O	计算
a	30	40	10
b	60	30	10
c	20	40	20

正确答案: A 你的答案: 空 (错误)

260,180
240,180
260,190
240,190

6 支笔，其笔身和笔帽颜色相同：但 6 支笔颜色各不相同，求全部笔身都戴错笔帽的可能性有多少种？

正确答案: A 你的答案: 空 (错误)

265
266
267
268

已知有序序列 b c d e f g q r s t,则在二分查找关键字 b 的过程中，先后进行比较的关键字依次是多少?()

正确答案: B 你的答案: 空 (错误)

f d b
f c b
g c b
g d b

如果待排序的数组已经近似递增排序，则此时快排算法的时间复杂度为()

正确答案: B 你的答案: 空 (错误)

O(n)
O(n^2)
O(nlogn)

O((n^2)*logn)

1000 以内与 105 互质的偶数有多少个?

正确答案: C 你的答案: 空 (错误)

227
228
229
230

函数 x 的定义如下,问 x(x(8)需要调用几次函数 x(int n)?

```
1    int x(int n){
2            cnt++;
3            if (n<=3)
4            {
5                    return 1;
6            }
7            else
8            {
9                    return x(n-2)+x(n-4)+1;
10           }
11   }
```

正确答案: B 你的答案: 空 (错误)

16
18
20
22

2015!后面有几个 0?

正确答案: C 你的答案: 空 (错误)

500
501
502
503

输入一个字符串,要求输出字符串中字符所有的排列,例如输入"abc",得到"abc","acb","bca","bac","cab","cba"

//递归实现，30 行，clean

```
#include<iostream>
```

```
#include<vector>
#include<string>
using namespace std;

vector<string> result;

void permute(string& str, int depth, int n){
    if(depth == n){
        result.push_back(str);
        return ;
    }
    for(int i = depth; i< n; i++){
        swap(str[depth],str[i]);
        permute(str, depth+1, n);
        swap(str[depth],str[i]);
    }
}

int main(){
    string str;
    cin>>str;
    permute(str, 0, str.size());
    for(int i = 0; i < result.size(); i++){
        cout << result[i] << endl;
    }
    return  0;
}
```

编写一个程序,将小于 n 的所有质数找出来。

```
1    #include <iostream>
```

```cpp
#include <cmath>
#include <vector>

using namespace std;

bool isprime(int x)
{
        if (x <= 1)  return false;
        else if (x == 2)  return true;

        for (int i = 2; i <=  sqrt(x); ++i)
        {
                if (x % i == 0)  return false;
        }

        return true;
}

vector<int> getAllPrimes(int n)
{
        vector<int> res;
        if (n < 2)  return res;

        for (int i = 2; i < n; ++i)
        {
                if (isprime(i))
                        res.push_back(i);
        }

        return res;
}
int main(void)
{
        int n;
        cin >> n;
        vector<int> prms = getAllPrimes(n);

        for (auto p : prms)
                cout << p << " ";
        cout << endl;

        return 0;
}
```

在一次活动中,我们需要按可控的比例来随机发放我们的奖品,假设需要随机的物品 id 和概率都在给定的 Map<String,Double>prizeMap 中,请实现如下这个函数: String getRandomPrize(Map<String,Double>prizeMap){}使得返回的结果为参与者 即将得到的一个随机物品 id.

prizeMap 中的数据为:

物品 id	投放概率
1	0.5
2	0.3
3	0.1
4	0.95
5.	0.05

```
#include <iostream>
#include <map>
#include <string>
#include <ctime>

using namespace std;

string getRandomPrize(map<string,  double> hm)
{
        double all = 0.0;

        map<double, string> mp;
        map<string,  double>::iterator it = hm.begin();
        while (it != hm.end())
        {
                all += (*it).second;
                mp.insert(pair<double, string>(all, (*it).first));
                ++it;
        }

        srand((unsigned)time(NULL));
        double total =  rand() /  double(RAND_MAX) * all;

        map<double, string>::iterator mpit = mp.begin();
        while (mpit != mp.end())
        {
                if (total < (*mpit).first)
                        return (*mpit).second;
                ++mpit;
        }
```

```
31
32          return "";
33  }
34
35  int main(void)
36  {
37          map<string,  double> hm;
38          hm.insert(pair<string,  double>("1", 0.5));
39          hm.insert(pair<string,  double>("2", 0.3));
40          hm.insert(pair<string,  double>("3", 0.1));
41          hm.insert(pair<string,  double>("4", 0.95));
42          hm.insert(pair<string,  double>("5", 0.05));
43
44          while (1)
45                  cout << getRandomPrize(hm);
46
47          return 0;
48  }
```