请计算以下各段代码的算法复杂度。

X=1;

O(1)

for (i=1; i<=n; i++)  
　　    x++;

**O(n)**

1. x=a;

a=b;

b=x;

O(1)

1. for (i=1; i<=n; i++)  
   　　    for (j=1; j<=n; j++)  
   　　        x=x+1;

O(n^2)

sum=0；                 
    for(i=1;i<=n;i++)        
       for(j=1;j<=n;j++)   
         sum++；

O(n^2)

6.

for (i=0;i<n;i++){

for (j=I;j<n;j++){

if a[i]>a[j]

{ temp=a[i];

a[i]=a[j];

a[j]=temp;

}

}

}

7.  
  
   for (i=1;i<n;i++)  
   {   
       y=y+1;           
       for (j=0;j<=(2\*n);j++)      
          x++;                
   }

O(n\*(1+(2\*n)\*(1)))=O(n^2))  
8.  
                                         
   a=0;  
   b=1;                       
   for ( i=1;i<=n; i++)   
   {    
      s=a +b;   
      b=a;　　　　　    
      a=s;　　　　　  
   }

O(1+1+n\*(3))=O(n)                                                        
9.   
   i=1;         
   while (i<=n)  
      i=i\*2;

O(log(2, n))  
10.    
   for(i=0;i<n;i++)  
   {    
      for(j=0;j<i;j++)    
      {  
         for(k=0;k<j;k++)  
            x=x+2;    
      }  
   }

**第二部分、算法练习---枚举法**

（一）

1. 什么是枚举法？特点？

**从所有的候选答案中逐一取出候选答案验证以后，找出正确的解（或无解）。一般使用多重循环，求解范围有边界，可预先确定答案的数量。**

1. 枚举法的一般步骤？

确定求解范围，设置验证条件，循环枚举。

1. 枚举法应用场景

问题可预先确定候选答案的数量，且候选答案的范围在求解之前有一个确定的集合

1. 枚举法适用局限

速度慢，计算量大

（二）编程或描述出以下各问题的解决方法

1．一个奇异三位数

一个自然数的七进制表达式是一个三位数，而这个自然数的九进制表示也是一个三位数，且这两个三位数的数码顺序正好相反，求这个三位数。

**def ten2**(*number*, *base*):  
 *"""进制转换"""* result = []  
 **while** *number*:  
 number, a = divmod(*number*, *base*)  
 result.append(str(a))  
 result.reverse()  
 **return** ''.join(result)  
  
  
**for** i **in** range(100, 1000):  
 **if** ten2(i, 7) == ten2(i, 9)[::-1]: # 数码顺序正好相反  
 print(i)  
  
# i = 248

2．

从键盘上任意输入一个正整数，如何判断该正整数是否是另一个整数的平方。

**import** math  
  
number = int(input("输入一个整数"))  
  
**for** i **in** range(int(math.sqrt(number)) + 1):  
 **if** i \* i == number:  
 print("输入的{}是{}的平方".format(number, i))  
 **break  
else**:  
 print("输入的{}不是另一个数的平方".format(number))

3．猜数游戏：

有一个4位的整数，并满足以下条件：

1） 数的前两位数字是相同的；

2）数的后两位数字也是相同的，但与前两位不同；

3）此数刚好是一个整数的平方。

请根据以上条件尝试找出这个数是多少。

**from** math **import** sqrt  
  
**for** i **in** range(1000, 10000): # 4位数  
 first2 = set(str(i)[:2]) # 前两个  
 **if** len(first2) == 1:  
 last2 = set(str(i)[2:]) # 后两个  
 **if** len(last2) == 1 **and not** last2 & first2: # 首尾相等  
 **if** int(sqrt(i)) \*\* 2 == i: # 验证是一个数的平方  
 print(i)  
# 7744

1. 设有一个数 X，当 X 加上100后得到的数 Y 是一个正整数的平方，然后用 Y 再加上68，又是另外一个正整数的平方，请问，如何找出 X 是多少?

**from** math **import** sqrt  
  
y1 = 10 # sqrt(y)  
**while** 1: # 存在解  
 y = y1 \* y1  
 x = y - 100  
 z = y + 68  
 **if** int(sqrt(z)) \*\* 2 == z: # 验证是一个数的平方  
 print(x)  
 **break  
 else**:  
 y1 += 1  
# x = 156

1. 若一个口袋中放有 12 个球，其中有 3 个红的， 3 个白的和 6 个黑的，问从中任取 8 个共有多少种不同的颜色搭配？

pocket = set() # 0:Red 1:White 2:Black  
**for** i **in** range(4):  
 **for** j **in** range(4):  
 **for** k **in** range(7):  
 **if** i + j + k == 8: # 8个球  
 pocket.add((i, j, k))  
**else**:  
 print("Total %d" % len(pocket))  
 **for** item **in** pocket:  
 print("Red:{}, White:{}, Black:{}".format(\*item))  
# Total 13  
# Red:3, White:3, Black:2  
# Red:1, White:1, Black:6  
# Red:0, White:2, Black:6  
# Red:0, White:3, Black:5  
# Red:2, White:2, Black:4  
# Red:2, White:0, Black:6  
# Red:2, White:1, Black:5  
# Red:2, White:3, Black:3  
# Red:3, White:1, Black:4  
# Red:3, White:0, Black:5  
# Red:1, White:3, Black:4  
# Red:3, White:2, Black:3  
# Red:1, White:2, Black:5