浙江财经大学
Zhejiang University Of Finance & Economics

# 2020初赛讲解

信智学院　陈琰宏

# 1 单项选择题

1.在内存储器中每个存储单元都被赋予一个唯一的序号，称为（A ）。

A. 地址　　B. 序号　　C. 下标　　D. 编号

2.编译器的主要功能是（A ）。

A. 将源程序翻译成机器指令代码

B. 将源程序重新组合

C. 将低级语言翻译成高级语言

D. 将一种高级语言翻译成另一种高级语言

3.设 x=true,y=true,z=false，以下逻辑运算表达式值为真的是（ D）。

A. (y∨z)∧x∧z　　B. x∧(z∨y) ∧z

C. (x∧y) ∧z　　D. (x∧y)∨(z∨x)

4.现有一张分辨率为 2048×1024 像素的 32 位真彩色图像。请问要存储这张图像，需要多大的存储空间？（C ）。

A. 16MB　　B. 4MB　　C. 8MB　　D. 2MB

2048*1024*32/8

# 1 单项选择题

5.冒泡排序算法的伪代码如下：
输入：数组L，n ≥ k。输出：按非递减顺序排序的 L。
算法 BubbleSort：

```
FLAG ← n //标记被交换的最后元素位置
  while FLAG > 1 do
      k ← FLAG -1
      FLAG ← 1
      for j=1 to k do
          if L(j) > L(j+1) then do
                L(j)  ↔ L(j+1)
                FLAG ← j
```

对 n 个数用以上冒泡排序算法进行排序，最少需要比较多少次?
（ C）。
A. n^2　　B. n-2　　C. n-1　　D. n
本题用flag记录下本轮是否有交换，如果没有 交换则循环结束。假设序列已经非递减顺序排序，进行n-1次比较后结束。

# 1 单项选择题

6.设 A是 n 个实数的数组，考虑下面的递归算法：

```
XYZ (A[1..n])
 if n=1 then return A[1]
 else temp ← XYZ (A[1..n-1])
      if temp < A[n]
       then return temp
       else return A[n]
```

请问算法 XYZ 的输出是什么？（B）。

A. A 数组的平均　　B. A 数组的最小值

C. A 数组的中值　　D. A 数组的最大值

n=1时返回A[1],否则返回小的这个数，xyz(a)=min(xyz(n-1),a[n])

7.链表不具有的特点是（A）。

A. 可随机访问任一元素

B. 不必事先估计存储空间

C. 插入删除不需要移动元素

D. 所需空间与线性表长度成正比

# 1 单项选择题

8. 有 10 个顶点的无向图至少应该有（A ）条边才能确保是一个连通图。

A. 9　　B. 10　　C. 11　　D. 12

最小生成树就可以保证联通，n个顶点n-1条边

9. 二进制数 1011 转换成十进制数是（ A ）。

A. 11　　B. 10　　C. 13　　D. 12

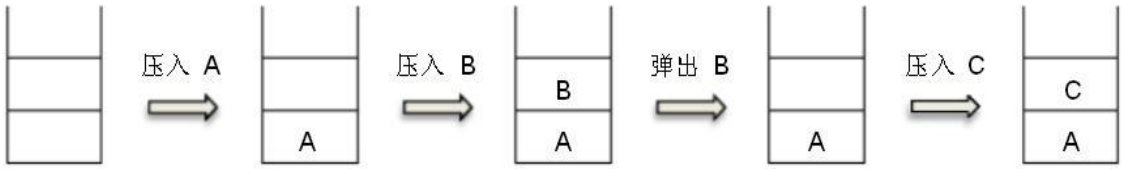
10. 5 个小朋友并排站成一列，其中有两个小朋友是双胞胎，如果要求这两个双胞胎必须相邻，则有（ A ）种不同排列方法?

A. 48　　B. 36　　C. 24　　D. 72

A(4,4)*A(2,2)=48

11. 下图中所使用的数据结构是（ A ）。

A. 栈　　B. 队列　　C. 二叉树　　D. 哈希表



# 1 单项选择题

12. 独根树的高度为 1。具有 61 个结点的完全二叉树的高度为（D ）。

A. 7　　B. 8　　C. 5　　D. 6　　$2^5<61<2^6$

13. 干支纪年法是中国传统的纪年方法，由 10 个天干和 12 个地支组合成 60 个天干地支。由公历年份可以根据以下公式和表格换算出对应的天干地支。

天干 =（公历年份）除以 10 所得余数

地支 =（公历年份）除以 12 所得余数

例如，今年是 2020 年，2020 除以 10 余数为 0，查表为"庚”；2020 除以 12，余数为 4，查表为“子” 所以今年是庚子年。

请问 1949 年的天干地支是（C ）

A. 己酉　　B. 己亥　　C. 己丑　　D. 己卯

1949%10=9

1949%12=5

| 天干 | 甲 | 乙 | 丙 | 丁 | 戊 | 己 | 庚 | 辛 | 壬 | 癸 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | | |
| 地支 | 子 | 丑 | 寅 | 卯 | 辰 | 巳 | 午 | 未 | 申 | 酉 | 戌 | 亥 |
| | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 0 | 1 | 2 | 3 |

# 1 单项选择题

14.10 个三好学生名额分配到 7 个班级，每个班级至少有一个名额，一共有（A ）种不同的分配方案。

A. 84　　B. 72　　C. 56　　D. 504

每个班至少一个名额，同时 （1,3，6）与（1,6,3）属于不同的分配方式，因此用隔板法就可以。9个位置分成6份，没份不为空。

C(9,6)=C(9,3)=84

15.有五副不同颜色的手套（共 10 只手套，每副手套左右手各 1 只），一次性从中取 6 只手套，请问恰好能配成两副手套的不同取法有（A）种。

A. 120　　B. 180　　C. 150　　D. 30

首先从5副手套中选出2副手套，C(5,2)，接着从剩下的6只中选择4次C(6,4),减去3中剩下两只手套也成对的情况，也就是

C(5,2)*(C(6,4)-3)=10*(15-3)=120

# 2 阅读程序写结果1[6328]

```
int main() {
  int k = 0;
  for (int i = 0; i < 26; ++i)
    if (encoder[i] != 0) ++k;
  for (char x ='A'; x <= 'Z'; ++x) {
    bool flag = true;
    for (int i = 0; i < 26; ++i)
      if (encoder[i] ==x) {
        flag = false;
        break;
    }
    if (flag) {
        encoder[k]= x;
        ++k;
    }
  }
  for (int i = 0; i < 26; ++i)
     decoder[encoder[i]- 'A'] = i + 'A';
  cin >> st;
  for (int i = 0; i < st.length( ); ++i)
    st[i] = decoder[st[i] -'A'];
  cout << st;
  return 0;
}
```

```
#include <cstdlib>
#include <iostream>
using namespace std;

char encoder[26] = {'C','S','P',0};
char decoder[26];

string st;

```

# 求得encoder数组

```
10 int main() {
11   int k = 0;
12   for (int i = 0; i < 26; ++i)
13     if (encoder[i] != 0) ++k;
14   for (char x ='A'; x <= 'Z'; ++x) {
15     bool flag = true;
16     for (int i = 0; i < 26; ++i)
17       if (encoder[i] ==x) {
18         flag = false;
19         break;
20     }
21     if (flag) {
22         encoder[k]= x;
23         ++k;
24     }
25   }
```

```
1 #include <cstdlib>
2 #include <iostream>
3 using namespace std;
4
5 char encoder[26] = {'C','S','P',0};
6 char decoder[26];
7
8 string st;
9
```

12-13行将初始的 C S P 负责给encoder, 14~25行，16-20行遍历查询26个字母，如果当前字母在encoder中没有出现过，则顺序赋值。出现过则下一个。形成如下所示的cnecoder数组

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| encoder | C | S | P | A | B | D | E | F | G | H | I | J | K | L | M | N | O | Q | R | T | U | V | W | X | Y | Z |
| 与‘A’相差 | 2 | 18 | 15 | 0 | 1 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 16 | 17 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |

# 求得decoder数组

```
26    for (int i = 0; i < 26; ++i)
27        decoder[encoder[i]- 'A'] = i + 'A';
28    cin >> st;
29    for (int i = 0; i < st.length( ); ++i)
30      st[i] = decoder[st[i] -'A'];
31    cout << st;
32    return 0;
33  }
```

```
1  #include <cstdlib>
2  #include <iostream>
3  using namespace std;
4
5  char encoder[26] = {'C','S','P',0};
6  char decoder[26];
7
8  string st;
9
```

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| encoder | C | S | P | A | B | D | E | F | G | H | I | J | K | L | M | N | O | Q | R | T | U | V | W | X | Y | Z |
| 与‘A’相差 | 2 | 18 | 15 | 0 | 1 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 16 | 17 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| decoder | D | E | A | F | G | H | I | J | K | L | M | N | O | P | Q | C | R | S | B | T | U | V | W | X | Y | Z |
| | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 英语字母表 | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |

# 2 阅读程序写结果1

判断题

(1)输入的字符串应当只由大写字母组成，否则在访问数组时可能越界。（ T ）

第 30 行 st[i]=decoder[ st[i]-A'],如果字符串 st 输入的数中不是大写字母,会导致 st[i]-'A的值大于或小于 0。

(2)若输入的字符串不是空串，则输入的字符串与输出的字符串一定不一样(F)

可以发现 如果 st[]字符串取“XY”的话,是一模-样的。

(3)将第 12 行的 i < 26 改为 i < 16，程序运行结果不会改变。（ T ）

因为 encoder 数组只有前三个进行了初始化,26 或者 16 并不会影响 k的值。

(4)将第 26 行的 i < 26 改为 i < 16，程序运行结果不会改变。（ F ）

第 26 行要遍历所有的 encoder 数组，产生 decoder 数组,改成 16 的话没办法把 decoder 数组中存入'A' ~'Z'

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| encoder | C | S | P | A | B | D | E | F | G | H | I | J | K | L | M | N | O | Q | R | T | U | V | W | X | Y | Z |
| 与‘A’相差 | 2 | 18 | 15 | 0 | 1 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 16 | 17 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| decoder | D | E | A | F | G | H | I | J | K | L | M | N | O | P | Q | C | R | S | B | T | U | V | W | X | Y | Z |
| 英语字母表 | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |

5） 若输出的字符串为 ABCABCABCA，则下列说法正确的是（ A ）。

A. 输入的字符串中既有 S 又有 P　　B. 输入的字符串中既有 S 又有 B

C. 输入的字符串中既有 A 又有 P　　D. 输入的字符串中既有 A 又有 B

根据英语字母表和 decoder 的对应可以查到若输出 ABCABCABCA,则输入为 CSPCSPCSPC.故既有 S 又有 P。

6）若输出的字符串为 CSPCSPCSPCSP，则下列说法正确的是（ D ）。

A. 输入的字符串中既有 P 又有 K　　B. 输入的字符串中既有 J 又有 R

C. 输入的字符串中既有 J 又有 K　　D. 输入的字符串中既有 P 又有 R

根据英语字母表和 decoder 的对应可以查到若输出 CSPCSPCSPCSP,则输入为PRNPRNPRNPRN,故既有 P 又有 R。

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| encoder | C | S | P | A | B | D | E | F | G | H | I | J | K | L | M | N | O | Q | R | T | U | V | W | X | Y | Z |
| 与'A'相差 | 2 | 18 | 15 | 0 | 1 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 16 | 17 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| decoder | D | E | A | F | G | H | I | J | K | L | M | N | O | P | Q | C | R | S | B | T | U | V | W | X | Y | Z |
| 英语字母表 | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |

# 2 阅读程序写结果2[6342]

输入入一个整数n和k，将n转化为k进制，同时输出从1开始累加到n，产生的进位次数。

```
#include <iostream>
using namespace std;
long long n, ans;
int k, len; //len表示位数，
long long d[1000000];
//d[i] 表示将 n转化为k位后的每一位的值
//n=10 d[i] 0101 len=4
//n=7  d[i] 111  len=3
```

```
int main() {
  cin >> n >> k;
  d[0] = 0;
  len= 1;
  ans = 0;
  for (long long i = 0; i <n; ++i) {
    ++d[0];
    for (int j = 0; j + 1<len; ++j) {
      if (d[j] == k) {//向高位进一位
        d[j] = 0;//当前位清0
        d[j + 1] += 1;//高位进1
        ++ans;// 产生一次进位
      }
    }
    if (d[len- 1] == k) {//最高位
      d[len - 1] = 0;
      d[len] =1;
      ++len;
      ++ans;
    }
  }
  for(int i=len-1;i>=0;i--)
    cout<<d[i];
  cout<<endl;
  cout << ans << endl;
  return 0;
}
```

判断题

(1)若 k=1，则输出 ans 时，len=n。　　（ F ）

当n=0,k=1时,模拟得到 len=1.

(2)若 k>1，则输出 ans 时，len 一定小于 n。　　（ F ）

当n=1,k=2时,模拟得到 len=1.

(3)若 k>1，则输出 ans 时，$k^{len}$一定大于 n。　　（ T ）

当n取得满足条件时的最大值，其每一位都为 k-1。所以 $n=k^{len}-1<k^{len}$。

所以对于任意的n都满足 $n<k^{len}$单选题

单选题

(4)若输入的 n 等于：$10^{15}$，输入的 k 为 1，则输出等于（ D）。

A. 1　　B. $(10^{30}-10^{15})/2$　　C. $(10^{30}+10^{15})/2$　　D. $10^{15}$

将$10^{15}$转化为一进制,每循环一次 ans 就会增加一次，故答案为$10^{15}$

(5)若输入的 n 等于 205,891,132,094,649（即 $3^{30}$ ），输入的 k 为 3，则输出等于（B ）。

A. $3^{30}$　　B. $(3^{30}-1)/2$　　C. $3^{30}-1$　　D. $(3^{30}+1)/2$

从低位到最高位:第一位 ans 增加$3^{29}$次，第二位 ans 增加$3^{28}$次.....，最高位 ans 增加$3^{0}$次,根据等比数列求和公式可得答案为$(3^{30}-1)/2$

(6)若输入的 n等于 100,010,002,000,090，输入的 k 为 10，则输出等于（D ）。

A. 11,112,222,444,543　　B. 11,122,222,444,453

C. 11,122,222,444,543　　D. 11,112,222,444,453

考虑更为一般的进位，也就是把ni 算出来,算出来为

# 2 阅读程序写结果3[6344]

```
#include <algorithm>
#include <iostream>
using namespace std;
int n;
int d[50][2];
int ans;
void dfs(int n, int sum) {                    ... }

int main() {
  cin >> n;
  for (int i = 0; i < n; ++i)
  cin >> d[i][0];
  for (int i = 0; i < n;++i)
     cin >> d[i][1];
  ans = 0;
  dfs(n, 0);
  cout << ans << endl;
  return 0;
}
```

给定两个序列A、B 两个序列，每次搜索 s=A 序列前两个数的和 + B 序列前两数差的绝对值，然后再把每个数列的两个合并，再进行下一次搜索。贪心求数列从头到尾依次合并。

```
 7 void dfs(int n, int sum) {
 8   if (n == 1) {
 9     ans = max(sum, ans);
10     return;
11   }
12   for (int i = 1; i < n; ++i) {
13     int a = d[i - 1][0];
14     int b = d[i - 1][1];
15     int x = d[i][0];
16     int y = d[i][1];
17     d[i - 1][0] = a + x;
18     d[i - 1][1] = b + y;
19     for (int j = i; j < n - 1; ++j)
20       d[j][0] = d[j + 1][0], d[j][1] = d[j + 1][1];
21     int s = a + x + abs(b - y);
22     dfs(n - 1, sum + s);
23     for (int j = n - 1; j > i; --j)
24       d[j][0] = d[j - 1][0], d[j][1] = d[j - 1][1];
25     d[i - 1][0] = a, d[i - 1][1] = b;
26     d[i][0] = x, d[i][1] = y;
27   }
28 }
```

程序分析: 输入两行数，每行n个数,每次可任选相邻的两个数进行合并 (相邻两个数相加,合并一次，　每行数字个数减一,同时统计累计答案, 累计规则是:　第l行合并的相邻两个数相加, 加上第2行对应两个数　相减的绝对值。　每行个数，所以一共可以合并n-次,在所有可能的　合并方案中求合并后的最大累计值。

假设n=5

| | i=[0] | i=[1] | i=[2] | i=[3] | i=[4] |
|---|---|---|---|---|---|
| d[i][0] | 4 | 2 | 7 | 9 | 6 |
| d[i][1] | 2 | 3 | 4 | 5 | 6 |

第1种合并可能

[0]和[1]合并，合并后在[0]，右侧数字左移一格

| i=[0] | i=[1] | i=[2] | i=[3] |
|---|---|---|---|
| 6 | 7 | 9 | 6 |
| 5 | 4 | 5 | 6 |

累加：4+2+abs(2-3)=7

sum=7

[0]和[1]合并

| [0] | [1] | [2] |
|---|---|---|
| 13 | 9 | 6 |
| 9 | 5 | 6 |

累加：6+7+abs(5-4)=14

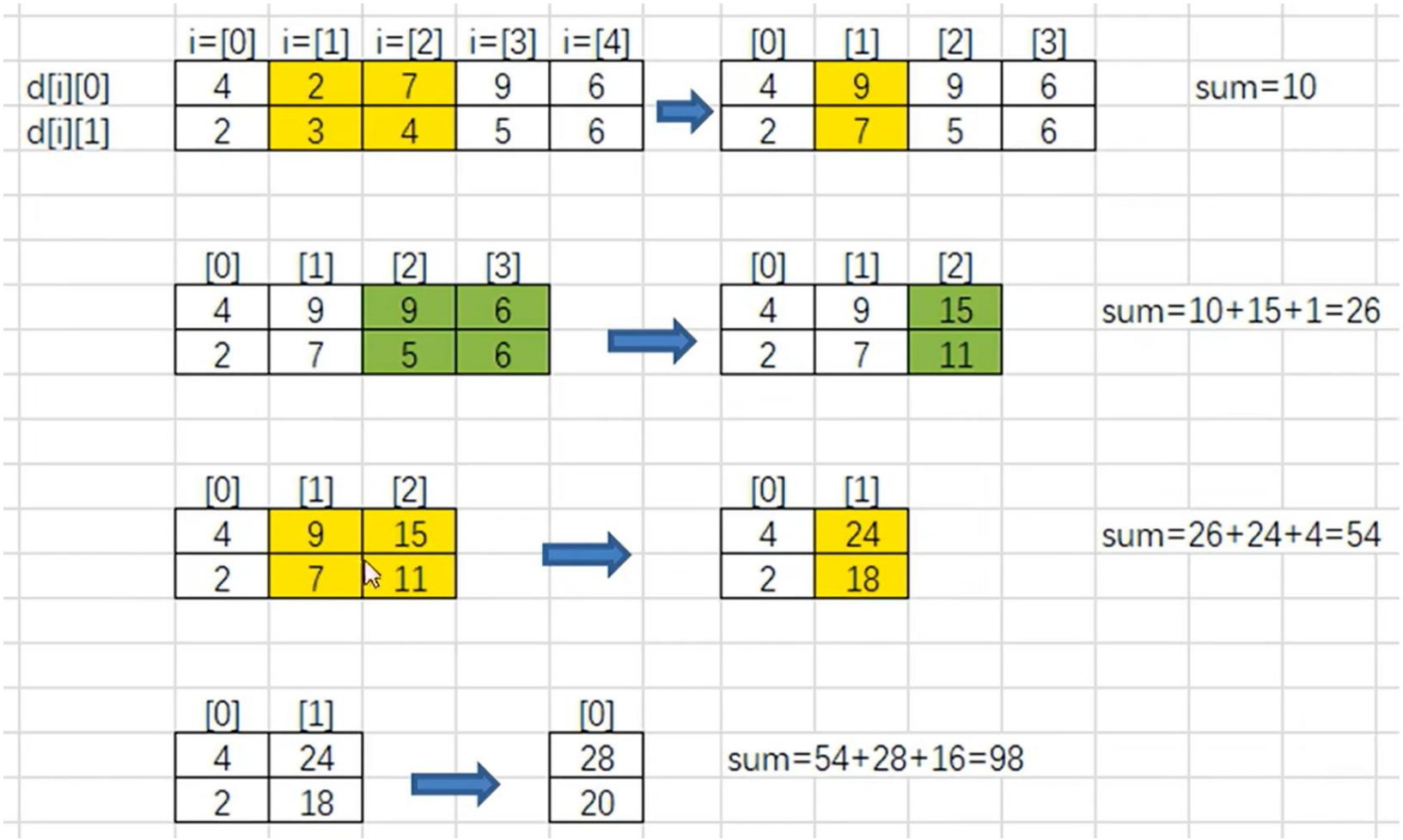
sum=21

[0]和[1]合并

| [0] | [1] |
|---|---|
| 22 | 6 |
| 14 | 6 |

累加：13+9+abs(9-5)=26

sum=47


i=[0] i=[1] i=[2] i=[3] i=[4]
d[i][0] 4 2 7 9 6
d[i][1] 2 3 4 5 6
[0] [1] [2] [3]
4 9 9 6
2 7 5 6
sum=10
[0] [1] [2] [3]
4 9 9 6
2 7 5 6
[0] [1] [2]
4 9 15
2 7 11
sum=10+15+1=26
[0] [1] [2]
4 9 15
2 7 11
[0] [1]
4 24
2 18
sum=26+24+4=54
[0] [1]
4 24
2 18
[0]
28
20
sum=54+28+16=98

```
05 int n;
06 int d[50][2];
07 int ans;
08
09 void dfs(int n, int sum) {
10   if (n == 1) {
11     ans = max(sum, ans);
12     return;
13   }
14   for (int i = 1; i < n; ++i) {
15     int a = d[i - 1][0], b = d[i - 1][1];
16     int x = d[i][0], y = d[i][1];
17     d[i - 1][0] = a + x;
18     d[i - 1][1] = b + y;
19     for (int j = i; j < n - 1; ++j)
20       d[j][0] = d[j + 1][0], d[j][1] = d[j + 1][1];
21     int s = a + x + abs(b - y);
22     dfs(n - 1, sum + s);
23     for (int j = n - 1; j > i; --j)
24       d[j][0] = d[j - 1][0], d[j][1] = d[j - 1][1];
25     d[i - 1][0] = a, d[i - 1][1] = b;
26     d[i][0] = x, d[i][1] = y;
27   }
```

3. d[][]全局变量

1. 递归返回点，元素个数是1个。到底滚动求最大值

4. 每次递归，局部变量n，i循环从[1]到[n-1]

5. 每次i循环，d[i-1][]和d[i][]暂存

6. 每次i循环，d[i-1][]更新为d[i-1][]+d[i][]

8. j循环，d[i+1][]到d[n-1]左移一格。d数组[0,n-1]变成[0,n-2]，元素个数变n-1

2. 仅递归调用1次，每次调用局部变量n减一。所以，递归n-1层。累计求和sum

9. 每次i循环，d[i+1][]到d[n-1][]恢复

1)若输入 n 为 0，此程序可能会死循环或发生运行错误。（ F ）

调用dfs函数的时候， if(n==1）,函数返回点不进去。 作循环不满足条件, 不进去，所以直接结束返回。不会死循环或发生运行错误。

(2)若输入 n 为 20，接下来的输入全为 0，则输出为 0。 （ T ）

(3)输出的数一定不小于输入的 d[i][0] 和 d[i][1] 的任意一个。（ F ）

反例。 2, d[0][0]=d[0][0]=1, d[0][1]=d[1][1]=1,输出2。

单选题

(4)若输入的 n 为 20，接下来的输入是 20 个 9 和 20 个 0，则输出为（B ）。

A. 1890 B. 1881 C. 1908 D. 1917

分析: 每层递归都合并[0]和[1],得到最大值。

2*9+3*9+4*++20*9=1881

因为di[1]=0所以相当于20个9,每次相令合并,合并的两个数的和累加,求最大值。

(5)若输入的 n 为 30，接下来的输入是 30 个 0 和 30 个 5，则输出为（C ）。
A. 2000　　B. 2010　　C. 2030　　D. 2020

0+5+10+15+28*5=29*14*5=2030
分析: 同上题，每层递归都合并[0]和[1],得到最大值。

(6)若输入的 n 为 15，接下来的输入是 15 到 1，以及 15 到 1，则输出为（ C ）。
A. 2440　　B. 2220　　C. 2240　　D. 2420

| | 累计 | | [0] | [1] | [2] | [3] | [4] | [5] | [6] | [7] | [8] | [9] | [10] | [11] | [12] | [13] | [14] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | d[i][0] | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |
| | | d[i][1] | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |
| d[ ][0] | 15+14 | | | 29 | 13 | | | | | | | | | | | | |
| d[ ][1] | 15-14 | | | 29 | 13 | | | | | | | | | | | | |
| d[ ][0] | 15+14+13 | | | | 42 | 12 | | | | | | | | | | | |
| d[ ][1] | 15+14-13 | | | | 42 | 12 | | | | | | | | | | | |
| d[ ][0] | 15+14+13+12 | | | | | 54 | 11 | | | | | | | | | | |
| d[ ][1] | 15+14+13-12 | | | | | 54 | 11 | | | | | | | | | | |
| 推导: | d[ ][0]部分 | | (15+14)+(15+14+13)+(15+14+13+12)+...+(15+14+13+…+2+1) | | | | | | | | | | | | | | |
| | d[ ][1]部分 | | (15-14 )+(15+14-13)+(15+14+13-12)+…+(15+14+13+…+2-1) | | | | | | | | | | | | | | |
| | 相加消项 | =2倍的 | 1*2+2*3+3*4+…+13*14+14*15 | | | | | | | | | | | | | | |
| | | =2倍的 | 1120 | | | | | | | | | | | | | | |

# 3完善程序 1[6345] 因式分解

19. 质因数分解）给出正整数 n，请输出将 n 质因数分解的结果，结果从小到大输出。
例如：输入 n=120，程序应该输出 2 2 2 3 5，表示：120=2×2×2×3×5。输入保证 2≤n≤$10^9$
.提示：先从小到大枚举变量 i，然后用 i不停试除 n 来寻找所有的质因子。

```
1  #include <cstdio>
2  using namespace std;
3  int n, i;
4  int main() {
5    scanf("%d", &n);
6    for(i = ① ; ②  <=n; i ++){
7      ③ {
8        printf("%d ", i);
9        n = n / i;
10     }
11   }
12   if(④ )
13     printf("%d ", ⑤ );
14   return 0;
15 }
```

1）①处应填（ C ）

A. 1　　B. n-1　　C. 2　　D. 0

2）②处应填（ C ）

A. n/i　　B. n/(i*i)　　C. i*i　　D. i*i*i

3）③处应填（ C )

A. if(n%i==0)　　B. if(i*i<=n)　　C. while(n%i==0)　　D. while(i*i<=n)

4）④处应填（ A ）

A. n>1　　B. n<=1　　C. i<n/i　　D. i+i<=n

5）⑤处应填（ C ）

A. 2　　B. n/i　　C. n　　D. i

(1)因为2是最小的素数,所以从2开始。如果从1开始直接死循环

(2)枚举到sqrt(n)即可

(3)要把i这个质因子全部除光。

(4)如果n>1,那就说明还存在 n 这个大素数没有除干净

(5)因为n是质数,所以输出 n即可。

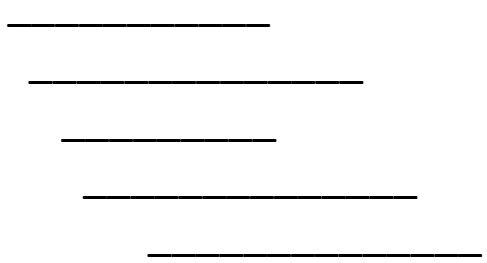
# 3完善程序 1[6346] 因式分解

19.（最小区间覆盖）给出 n 个区间，第 i 个区间的左右端点是 [ai,bi ]。现在要在这些区间中选出若干个，使得区间 [0,m] 被所选区间的并覆盖（即每一个 0≤i≤m 都在某个所选的区间中）。保证答案存在，求所选区间个数的最小值。
输入第一行包含两个整数 n 和 m (1≤n≤5000,1≤m≤$10^9$)
接下来 n 行，每行两个整数 ai ,bi（0≤ai,bi≤m）。
提示：使用贪心法解决这个问题。先用O($n^2$) 的时间复杂度排序，然后贪心选择这些区间。
试补全程序。

```
3  const int MAXN = 5000;
4  int n, m;
5  struct segment { int a, b; } A[MAXN];
6
7  void sort() // 排序
8  {//按照左端点从小到大排序
9      for (int i = 0; i < n; i++)
10     for (int j = 1; j < n; j++)
11     if ( (1) )
12     {
13         segment t = A[j];
14           (2)
15     }
16 }
17
18 int main()
19 {
20     cin >> n >> m;
21     for (int i = 0; i < n; i++)
22         cin >> A[i].a >> A[i].b;
23     sort();
24     int p = 1;
```

```
18 int main()
19 {
20     cin >> n >> m;
21     for (int i = 0; i < n; i++)
22         cin >> A[i].a >> A[i].b;
23     sort();
24     int p = 1;
25
26     for (int i = 1; i < n; i++)
27         if ( (3))
28             A[p++] = A[i];
29     n = p;//记录筛选后的线段数量
30
31     int ans =0, r = 0;
32     int q = 0;
33     while (r < m)
34     {
35         while (  (4)  )
36             q++;
37         (  5  )
38         ans++;
39     }
40     cout << ans << endl;
41     return 0;
42 }
```

贪心：
先将n个区间按照起点顺序从小到大排序，从排好序的第一个区间开始，令r表示覆盖到的区域[0,r],在剩下的区间中找出所有左端点小于等于当前已经覆盖区域的右端点，并且右端点大于等于之前覆盖区域的右端点r，
取右端点最大的区间加入已覆盖区域并赋值给r。直到r=m结束。



```cpp
#include <iostream>
using namespace std;
const int MAXN = 5000;
int n, m;
struct segment { int a, b; } A[MAXN];

void sort() // 排序
{//按照左端点从小到大排序
    for (int i = 0; i < n; i++)
    for (int j = 1; j < n; j++)
    if (A[j].a<A[j-1].a)
    {
        segment t = A[j];
        A[j]=A[j-1];A[j-1]=t;
    }
}

```

```
18 int main()
19 {
20     cin >> n >> m;
21     for (int i = 0; i < n; i++)
22         cin >> A[i].a >> A[i].b;
23     sort();
24     int p = 1;
25     //先筛选出可能的线段，即右端点都是递增的线段
26     for (int i = 1; i < n; i++)
27         if ( A[i].b>A[p-1].b)
28             A[p++] = A[i];
29     n = p;//记录筛选后的线段数量
30  /*
31  _____1
32      2'_____2
33          3'_____3
34              4'_____4
35
36  */
```

```
37      int ans =0, r = 0;
38      int q = 0;
39      while (r < m)
40      {
41  //找到左端点小于等于当前已覆盖区域的右端点，且右端点大于等于
42  //并当前已覆盖区域的右端点的区间，然后更新已覆盖区域的右端点
43          while (q+1<n&&A[q+1].a<=r)
44              q++;
45          r=max(r,A[q].b);
46          ans++;
47      }
48      cout << ans << endl;
49      return 0;
50  }
```

1)①处应填（ B ）

A. A[j].b>A[j-1].b　B. A[j].a<A[j-1].a　C. A[j].a>A[j-1].a　D. A[j].b<A[j-1].b

排序肯定先对左端点排序,再去调换位置

2)②处应填（ D ）//交换变量的基本操作。

A. A[j+1]=A[j];A[j]=t;　B. A[j-1]=A[j];A[j]=t;

C. A[j]=A[j+1];A[j+1]=t;　D. A[j]=A[j-1];A[j-1]=t;

3)③处应填（C ）

A. A[i].b>A[p-1].b　B. A[i].b<A[i-1].b　C. A[i].b>A[i-1].b　D. A[i].b<A[p-1].b

线段如果不包含了就应该加上,也就是当前的右端点比最后一个右端点大,第一个肯定在，如果包含那就没必要选择了。

4)④处应填（ B ）

A. q+1<n&&A[q+1].a<=r　B. q+1<n&&A[q+1].b<=r

C. q<n&&A[q].a<=r　D. q<n&&A[q].b<=r

根据题解方法,找到左端点小于等于当前已覆盖区域的右端点，并且右端点大于等于当前已覆盖区域的右端点的区间，然后更新已覆盖区间[0,r]的右端点。

5)⑤处应填（ C ）

A. r=max(r,A[q+1].b)　B. r=max(r,A[q].b)　C. r=max(r,A[q+1].a)　D. q++

2.先将n个区间按照起点顺序从小到大排序,从排好序的第一个区间开始，令r 表示已经覆盖到的区域ro,r],在剩下的区间中找出所有左端点小于等于当前已经覆盖区域的右端点，并且右端点大于等于之前覆盖区域的右端点r,取右端点最大的区间加人已覆盖区域并赋值给r。直到r=m 结束。
(1)排序肯定先对左端点排序,再去调换位置
(2)交换变量的基本操作。
(3)线段如果不包含了就应该加上,也就是当前的右端点比最后一个右端点大,第一个肯定在，如果包含那就没必要选择了。
(4)根据题解方法,找到左端点小于等于当前已覆盖区域的右端点，并且右端点大于等于当前已覆盖区域的右端点的区间，然后更新已覆盖区间[o,r]的右端点。
(5)r会被更新为当前的右端点

# 今天的课程结束啦……

**下课了…**
**同学们再见！**