一、单项选择题（共15题，每题2分，共计30分）

1	2	3	4	5	6	7	8	9	10
A	A	D	C	C	B	A	A	A	A
11	12	13	14	15					
A	D	C	A	A					

二、阅读程序（除特殊说明外，判断题1.5分，单选题3分，共计40分）

第1题	判断题（填√或×）				单选题	
	1)	2)	3)	4)	5)	6)
	√	×	√	×	A	D
第2题	判断题（填√或×）			单选题		
	1)	2)	3)	4)	5)	6)
	×	×	√	D	B	D
第3题	判断题（填√或×）			单选题		
	1)	2)	3)	4)	5)	6)（4分）
	×	√	×	B	C	C

三、完善程序（单选题，每小题3分，共计30分）

第1题					第2题				
1)	2)	3)	4)	5)	1)	2)	3)	4)	5)
C	C	C	A	C	B	D	A	A	B

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. 地址　B.序号　　C.下标　D.编号

正确答案：A

解析：变量的地址就是它在内存的位置

2.编译器的主要功能是()。

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

正确答案：A

解析：编译器的作用是把高级语言转换成计算机明白的汇编语言，故选 A。

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

A. (y∨z)∧x∧z B. x∧(z∨y)∧z C. (x∧y)∧z D. (x∧y)∨(z∨x)

正确答案:D

解析：在逻辑表达式里 v 代表或，^代表与，

x	y	z
1	1	0

A 选项的数值是(1v0)^1^0,所以数值是 0

B 选项的数值是 1^(0v1)^0,所以数值是 0

C 选项的数值是 (1^1)^0,所以数值是 0

D 选项的数值是(1^1)v(0v1),所以数值是 1

所以正确答案是 D。

4.现有一张分辨率为 2048x1024 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?()。

A.16MB B.4MB C.8MB D.32MB

正确答案：C

解析:2048 = 2^{11};1024 = 2^{10} 相乘的结果是 2^{21} 一共是个 2^{21} 像素

每个像素是 32 位一共 2^5 ，所以一共 2^{26} 个。

我们看看以 bit 为单位，各个计算机数值单位

	1B	1KB	1MB	1GB	1TB
Bit 数	2^3	2^{13}	2^{23}	2^{33}	2^{43}

所以这些一共是 8MB。正确答案是 C

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

```
1.    FLAG ← n // 标记被交换的最后元素位置
2.    while FLAG > 1 do
3.       k ← FLAG - 1
4.       FLAG ← 1
5.          for j=1 to k do
6.                if L (j) > L(j+1) then do
7.                      L(j) ↔ L(j+1)
8.                      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()

A n^2　B n-2　C.n-1　D.n

正确答案：C

答案解析:伪代码里←是赋值的意思。最少情况为当输入的数组按升序排列的情况。第 4 句将 FLAG 变为 1。while 循环的第一次，j 会从 1 循环到 n-1，第 6 句的比较共进行 n-1 次，但是没有任何一次比较能进入第 7、8 句那么 FLAG 就会保持为 1。这样 while 循环就不再成立了。共比较 n-1 次。

6.设 A 是和个实数的数组，考虑下面的递归算法:

XYZ (A[1..n])

1. 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?()

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

正确答案：B

解析:如果 n=1，答案为 a[1]，否则，答案和前 n-1 项的结果 XYZ(A[1..n-1])有关。如果 xYZ (A[1..n-1])是比 a[n]小的，XYZ(A[1..n])的答案为更小的 XYZ(A[1..n-1])，否则为 a[n]。这其实就是找数组 a 里的最小值。

7. 链表不具备的特点是()

A.可随机访问任意元素

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

正确答案：A

解析:链表访问某一点的元素需要从头结点开始一步步访问

8.有 10 个顶点的无向图至少应该有()条边才能确保是一个连通图。

A.9　　B. 10　　C.11　　D. 12

正确答案：A

解析:连通图的意思是所有的点都是可达的。最小连通图边的数量 = 点的数量-1。

9.二进制数 1011 转换成十进制数是()

A. 11　　B.10　　C.13　　D.12

正确答案：A

解析:(1011)2 的十进制法是用位值定理求，我们认为是

$2^3+2^1+2^0 = 8+2+1 = 11$。故选 A

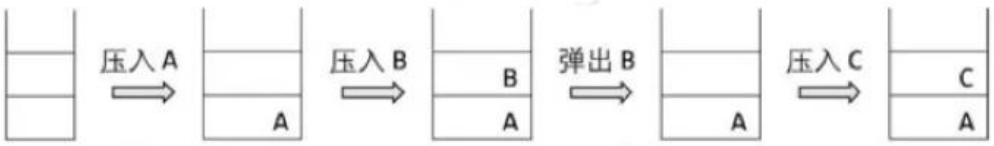
10.五个小朋友并排站成一列，其中有两个小朋友是双胞胎，如果要求这两个双胞胎必须相邻，则有(A)种不同排列方法?

A. 48　　B.36　　C.24　　D.72

正确答案：A

解析:我们用捆绑法把这两个双胞胎合在一起，看成一个人总共就是 4 个人，当然这两个双胞胎也需要排列，所以中的排列数根据乘法定理是 4!*2! = 24*2 = 48，故选 A。

11.下图中所使用的的数据结构是()



A. 栈　　B.队列　　C.二叉树　　D.哈希表

正确答案：A

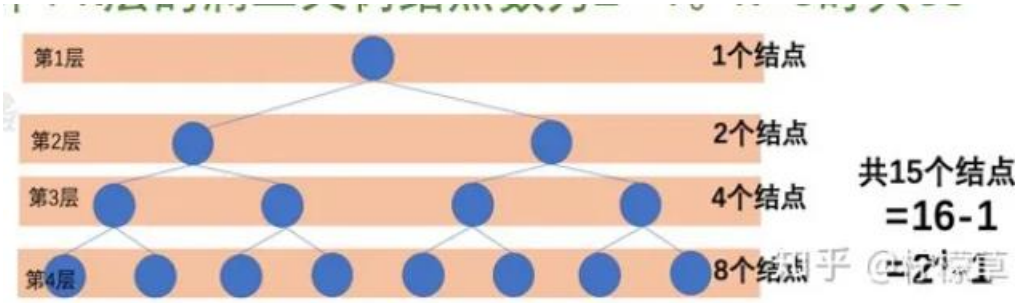
解析:栈是先进先出的数据结构，所以答案是 A。

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()

A. 7　　B. 8　　C.5　　D.6

正确答案：D

解析:k 层的满二叉树结点数为 2^k-1。k=6 时共 63 个结点。



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

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

例如，今年是 2020 年，2020 除以 10 余数为 0,查表为所以今“庚”;2020 除以 12,余数为 4,查表为“子”年是庚子年。请问 1949 年的天干地支是()

A. 己酉　　B.己亥　C.己丑　D.己卯

正确答案：D

解析:1949 除 10 余数为 9，查表为”己”，1949 除 12 余数为 5，查表为”丑”，所以是”己丑”年。故选 C。

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 ()种不同的分配方案。

A. 84　　B.72　C.56　D.504

正确答案：A

解析:枚举

1111114 此方案根据 4 的位置，共有 7 种.

1111123 此方案根据 2、3 的位置，共有 C^2_7=42 种

1111222 此方案根据 2 的位置，共有 C^3_5=35 种

共有 7+42+35=84 种

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()种。

A. 120　　B.180　　C.150　　D.30

正确答案：A

解析:配成两副手套，有 C^2_5=10 种方法。剩下的 2 只，需要 2 种不同的颜色，有 C^2_3;=3 种方法，取出时同一颜色根据取出是左是右又有两种情况，因此最终答案为 10*3*2*2=120 种方法。

阅读程序(程序输入不超过数组或字符串定义的范围;判断题正确填 √，错误填 x 除特殊说明外，判断题 1.5 分，选择题 3 分，共计 40 分)

```
01 #include <cstdlib>
02 #include <iostream>
03 using namespace std;
04
05 char encoder[26] = {'C', 'S', 'P', 0};
06 char decoder[26];
07
08 string st;
09
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    }
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  }
```

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。()

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。()

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

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

单选题

5)若输出的字符串为“ABCABCABCA”，则下列说法正确的是()。

A.输入的字符串中既有 A 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 S 又有 P

D.输入的字符串中既有 A 又有 B

6)若输出的字符串为“CSPCSPCSPCSP”，则下列说法正确的是()。

A.输入的字符串中既有 J 又有 R

B.输入的字符串中既有 P 又有 K

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

解析：本程序主要设计的是 encoder 数组跟 decoder 数组的关系，一开始 encoder 数组初始化为{C,S,P};

```
11   int k = 0;
12   for (int i = 0; i < 26; ++i)
13     if (encoder[i] != 0) ++k;
```

所以这一行代码，执行之后 k 的数值是 3

然后以下的代码是为了填充 encoder 数组

```
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   }
```

我们审问，char 型变量 x 从 A~Z，如果 x 没有在 encoder 数组里出现

```
16     for (int i = 0; i < 26; ++i)
17       if (encoder[i] == x) {
18         flag = false;
19         break;
```

那么 flag 的数值为 1，否则数值就是 0。

当 flag 为 1 时

```
21      if (flag) {
22        encoder[k] = x;
23        ++k;
24      }
```

我们就把这个未出现的 x 赋给 encoder 的新内容

所以这个 encoder 数组的数值是

i	A	B	C	D	E	F	G	H	I	J	K	L	M
e[i]	C	S	P	A	B	D	E	F	G	H	I	J	K
i	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
e[i]	L	M	N	O	Q	R	T	U	V	W	X	Y	Z

随后我们根据 encoder 填写 decoder 数组

```
26    for (int i = 0; i < 26; ++i)
27      decoder[encoder[i] - 'A'] = i + 'A';
```

我们发现是 encoder 跟 decoder 是反函数

i	C	S	P	A	B	D	E	F	G	H	I	J	K
d[i]	A	B	C	D	E	F	G	H	I	J	K	L	M
i	L	M	N	O	Q	R	T	U	V	W	X	Y	Z
d[i]	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

然后我们输入字符串 st，按照 decoder 序输出

```
28    cin >> st;
29    for (int i = 0; i < st.length(); ++i)
30      st[i] = decoder[st[i] - 'A'];
31    cout << st;
```

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。()

正确答案：正确

解析：decoder 数组容量只有 26 个，多余的会溢出

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。()

正确答案：错误

解析：字符串 st 所有字符在[T,Z]范围内，输入输出不变！！

T	U	V	W	X	Y	Z
T	U	V	W	X	Y	Z

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

5)若输出的字符串为“ABCABCABCA”，则下列说法正确的是()。

A.输入的字符串中既有 A 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 S 又有 P

D.输入的字符串中既有 A 又有 B

正确答案：正确

解析：第 12 行的位置是

```
12   for (int i = 0; i < 26; ++i)
13     if (encoder[i] != 0) ++k;
```

一开始 encoder 就三个字符{‘C’,’S’,’P’};所以就算 i<16 也不会出

现问题

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

正确答案：错误

解析：第 26 行的位置

```
26   for (int i = 0; i < 26; ++i)
27     decoder[encoder[i] - 'A'] = i + 'A';
```

这个是 decoder 数组赋值的位置，如果改成 16，会导致 decoder[15~25]的数值为 0,从而导致 decode 结果出现偏差的可能，所以错误。

5)若输出的字符串为“ABCABCABCA”，则下列说法正确的是()。

A.输入的字符串中既有 A 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 S 又有 P

D.输入的字符串中既有 A 又有 B

正确答案:C

解析:我们张开 decoder 表

i	C	S	P	A	B	D	E	F	G	H	I	J	K
d[i]	A	B	C	D	E	F	G	H	I	J	K	L	M
i	L	M	N	O	Q	R	T	U	V	W	X	Y	Z
d[i]	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

“ABCABCABCA”是输出的字符串

d[C] = A　d[S] = B d[P] = C,所以输入的一定是有 S,有 P

正确答案选 C

6)若输出的字符串为“CSPCSPCSPCSP”，则下列说法正确的是()。

A.输入的字符串中既有 J 又有 R

B.输入的字符串中既有 P 又有 K

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

正确答案:D

解析:我们张开 decoder 表

i	C	S	P	A	B	D	E	F	G	H	I	J	K
d[i]	A	B	C	D	E	F	G	H	I	J	K	L	M
i	L	M	N	O	Q	R	T	U	V	W	X	Y	Z
d[i]	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

“CSPCSPCSP”是输出的字符串,d[P] = C,d[N] = P, d[R] = S

有 P，R,所以正确答案选 D。

第 17 题

2.

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

long long n, ans;
int k, len;
long long d[1000000];

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) {
        d[j] = 0;
        d[j + 1] += 1;
        ++ans;
      }
    }
    if (d[len- 1] == k) {
      d[len - 1] = 0;
      d[len] =1;
      ++len;
      ++ans;
    }
  }
  cout << ans << endl;
  return 0;
}
```

假设输入的 n 是不超过 2^{62} 的正整数，k 都是不超过 10000 的正整数，

完成下面的判断题和单选题：

判断题

1. 若 $k=1$，则输出 ans 时，len=n。（ ）
2. 若 k>1，则输出 ans 时，len 一定小于 n。（ ）
3. 若 $k>1$，则输出 ans 时，k^{len}一定大于 n。（ ）

单选题

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

A. 1

B. $(10^{30}-10^{15})/2$

C. $(10^{30}+10^{15})/2$

D.10^{15}

2.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（ ）。

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

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

A.11,112,222,444,543

B.11,122,222,444,453

C.11,122,222,444,543

D.11,112,222,444,453

解析：我们先看一下这个程序的做法

我们对程序模拟一下 n = 5，k =3 对程序模拟一下

Len = 1，ans = 0，

我们的 d[0] == 3 了，触发第 22 行的

```
if (d[len- 1] == k) {
  d[len - 1] = 0;
  d[len] =1;
  ++len;
  ++ans;
```

变成了

1	0

然后 d[0]+=2;等于

1	2

也就是说 d 数组初步看是记录整数 n 的第 k 进制表示方式

len 代表 d 数组的长度

ans 代表了进位的次数

那么 ans 跟 n 到底什么关系呢？

我们关系最近的 n = 3969 来说吧

大家都是从 d[0]处往 d[1]处外溢的,且外溢了 3969/10 = 396

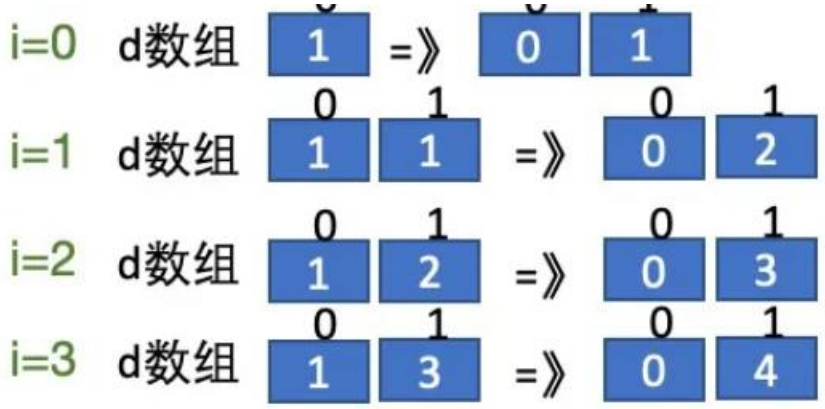
第 0 位	第 1 位	第 2 位	第 3 位
3969/10	396/10	39/10	3/10

判断题

若 k=1，则输出 ans 时，len=n。 （ ）

正确答案：错误

解析：我们试着输入 n = 4，k =1，所以不对



最后 len 不等于 n。

2. 若 k>1，则输出 ans 时，len 一定小于 n。 （ ）

正确答案:错误

解析:他的意思是 nk 进制表示的数的位数 len 一定小于 n 吗?

如果 n= 2 k=2，2 的二进制表示就是(10)2

那么这个 len 就是等于 2.相等！！

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

正确答案：正确

解析：一个 k 进制数，如果有 len 位，每一位有 k 种变化，那么一共能表示 k^{len}种数值，数值是从 0 到 $k^{len-1}-1$ 的。n 就是在 0 到 $k^{len-1}-1$ 之间的，所以可以说 k^{len}一定大于 n。

2．选择题

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

A. 1

B. $(10^{30}-10^{15})/2$

C. $(10^{30}+10^{15})/2$

D.10^{15}

正确答案： D

解析:根据判断题 1)的举例，可以发现，当 k=1 时每个 i 会发生一次进位，共循环 n 次，因此共发生 n 次进位即输出 n。

2.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（ ）。

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

正确答案： B

解析:根据上面的“解题思路”部分，ans 是指为了获得 n 共发生了多少次进位。如果是十进制的 100，共发生了 100/10+100/100 次进位，即 $n/10^1+n/10^2$ 次。同样，k=3 进制时，个位每 3 次都会进一位，因此 n 的个位会有 n/3 次进位。$n=3^{30}$ 时，共发生 $3^{30}/3+3^{30}/3^2+...+3^{30}/330=3^{29}+...+3^2+3+1=(1-3^{30})/(1-3)=(3^{30}-1)/2$。(根据等比级数求和的公式)

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

A.11,112,222,444,543

B.11,122,222,444,453

C.11,122,222,444,543

D.11,112,222,444,453

正确答案： D

解析:根据之前的分析，

个位共会发生 $n/10^1$ =10001000200009 次.

进位百位共会发生 $n/10^2$=1000100020000 次。

进位千位共会发生 $n/10^3$=100010002000 次进位

10001000200009
1000100020000
100010002000
10001000200
1000100020
100010002
10001000
1000100
100010
10001
1000
100
10
1

一共是 11，112，222，444，453 次进位

3.

```
01 #include <algorithm>
02 #include <iostream>
03 using namespace std;
04
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   }
28 }
29
30 int main() {
31   cin >> n;
32   for (int i = 0; i < n; ++i)
33     cin >> d[i][0];
34   for (int i = 0; i < n; ++i)
35     cin >> d[i][1];
36   ans = 0;
37   dfs(n, 0);
38   cout << ans << endl;
39   return 0;
40 }
```

假设输入的 n 是不超过 50 的正整数，d[i][0]、d[i][1]都是不超过 10000 的正整数，完成下面的判断题和单选题:

1.判断题

1)若输入 n 为 0 ，此程序可能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 。()

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()

单选题

4) 若输入的 n 为 28，接下来的输入是 20 个 9 和 20 个 6，则输出为()

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

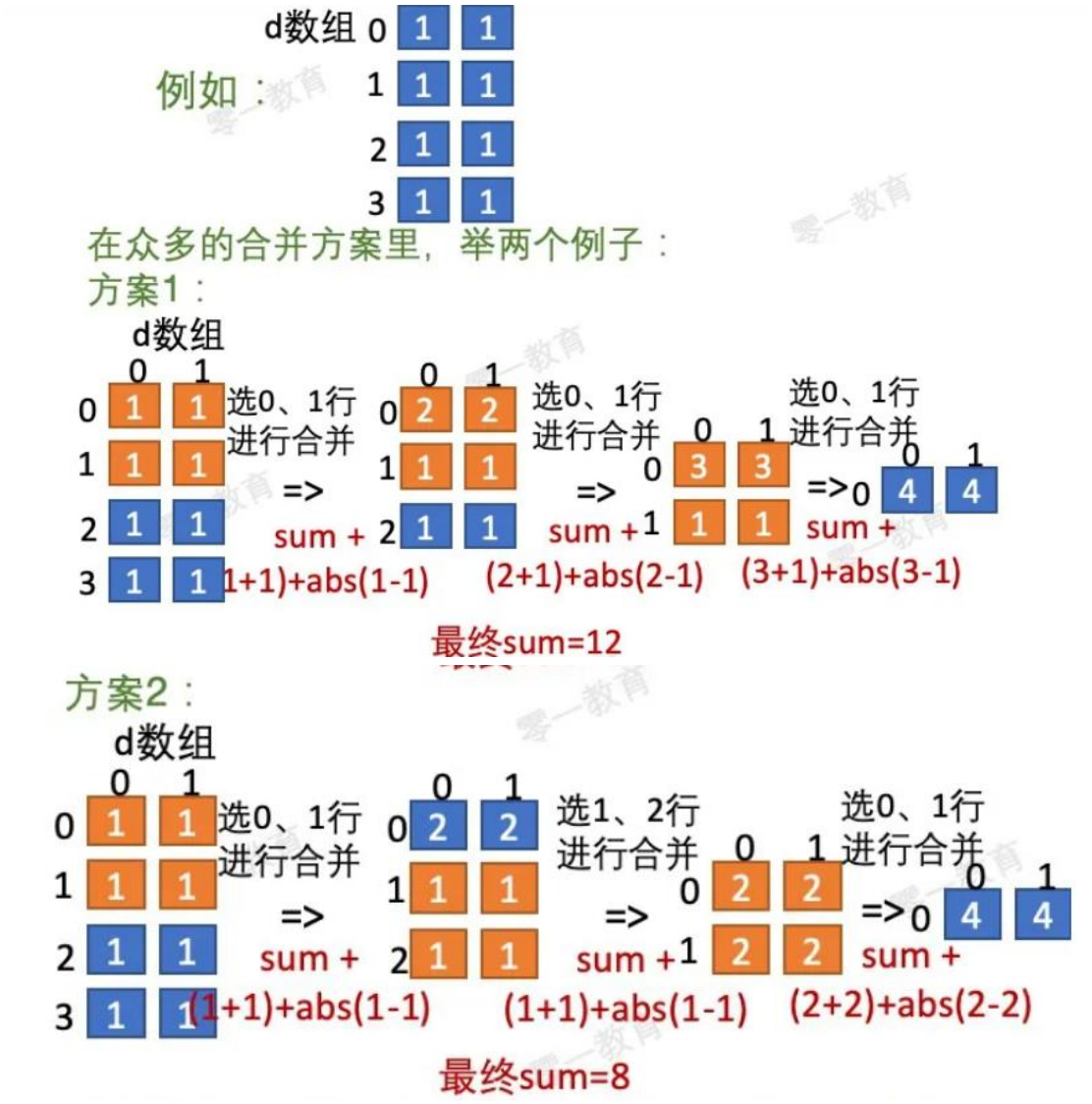
5) 若输入的 n 为 30，接下来的输入是 30 个 0 和 30 个 5，则输出为()(4 分)

A.2020 B.2030 C.2010 D.2000

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()

A.2420 B.2220 C.2440 D.2240

思路解析:根据题目描述，本题是针对一个有 2 列的二维数组，不断选出相邻的两行进行合并，直到合并成 1 行为止。结果为每次合并时【相邻两行第 0 列值之和+第 1 列之差的绝对值】，取加和。根据每次选哪两行会产生很多不同的方案，ans 为其中加和最大的方案。本题与信奥经典动规题目“石子合并有异曲同工之处。但是此题会用更多的贪心思想。



可以看到，不同的合并方案，最终 sum 是不一样的，方案 1 的 sum 要好于方案 2 的 sum。

判断题解析：

1)若输入 n 为 0，此程序可能会死循环或发生运行错误。()

正确答案：正确

解析:若 n=0，第 10 句和第 14 句都不成立，那么程序会立即返回主函数结束，不会发生错误。

2)若输入 n 为 20，接下来的输入全为 0 ，则输出为 0。()

正确答案：正确

解析:根据题目描述，这种情况每次的 sum 都只能加 0，最终 sum 为 0。

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()

正确答案：错误

答案解析:若输入 n=1，ans=0，是小于 d 数组的

4) 若输入的 n 为 28，接下来的输入是 20 个 9 和 20 个 6，则输出为()

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

正确答案： C

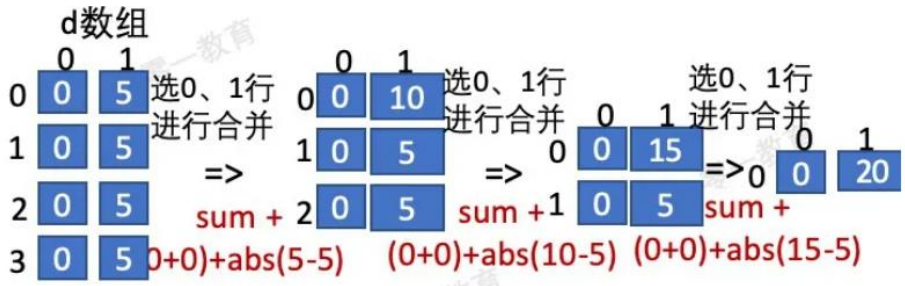
解析:这种情况，相当于忽略第 1 列，简化了程序。根据思路解析里的例子，运用贪心思想可以发现，每次让刚刚合并过的行继续合并，会得到更大的 sum 即像滚雪球一样越滚越大。因此可以用第 1 行与第 2 行合并，其合并的结果继续与下面一行合并…这样得到的结果为:(9+9)+(9+9+9)+(9+9+9+9)+ ..(9+9+...+9)。其中最后一项为 20 个 9 相加。答案为 9*2+9*3+...+9*20=9*(20*21/2-1)=1881

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()(4 分)

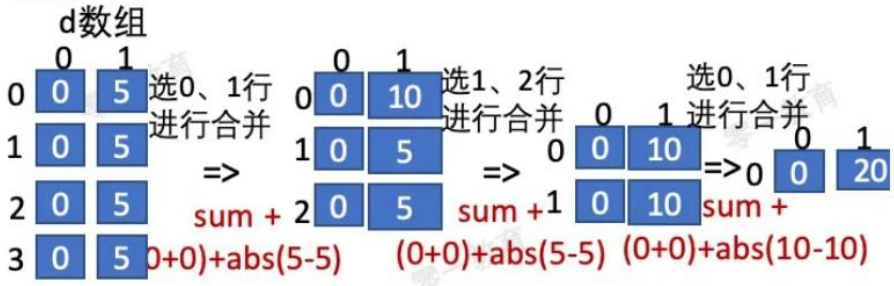
A.2020　B.2030　C.2010　D.2000

正确答案：B

解析:因为 n 比较大，不易看出规律，可以举一个 n 较小的例子，发现规律。例如 n=4，或者 n=6。



方案2：



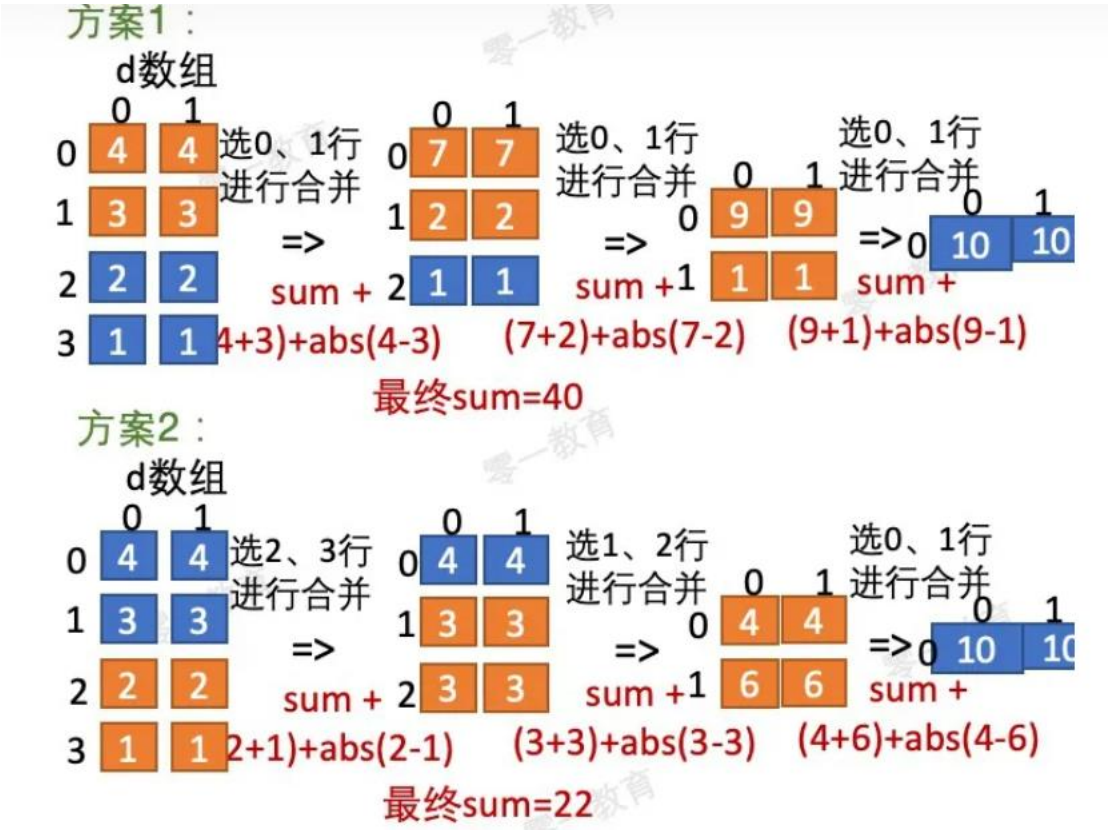
在上述两种方案里，方案 1 的结果更大。结果为 0+5+10.那么可以推广到 n=30 的情况，结果为 0+5+10+...+28*5=5*(28*29/2)=2030.(因为 n=4 时加到了 2*5，可以推出 n=30 时会加到 28*5

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()

A.2420　　B.2220　C.2440　D.2240

正确答案：

解析：因为 n 比较大，不易看出规律，可以举一个 n 较小的例子，发现规律。例如 n=4，或者 n=6。



在上述两种方案里，方案 1 的结果更大。sum 为:第 1 列的每次又加又减自动抵消，第 0 列对 sum 的贡献趋势为

2*(4)+

2*(4+3)+

2*(4+3+2).

可以看到 4 有 3 项，3 有 2 项，2 有 1 项:推广到 n=15 的情况:

答案为:2*(15*14+14*13+13*12+...+2*1)。根据数学定理1*2+2*3+3*4+4*5+．．．．+n(n+1)=n(n+1)(n+2)/3。因此答案为 2*14*15*16/3=2240

三、完善程序(单选题，每小题 3 分，共计 30 分)

1.（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 至大输出。
例如：输入n=120,程序应该输出2 2 2 3 5,表示120=2 X 2 X 2 X 3 X 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。

试补全程序。

```
01 #include <cstdio>
02 using namespace std;
03
04 int n, i;
05
06  int main() {
07      scanf(”%d“, &n);
08      for(i = ①;② <= n; i ++) {
09          ③{
10              printf("%d ", i);
11              n = n / i;
12          }
13      }
14      if (④)
15          printf("%d ", ⑤);
16      return 0;
17 }
```

1)①处应填()

A. n-1　　B. 0　　C．1　　D．2

2)②处应填()

A. n/i　B. n/(i*i)　　C．i*i　D. i＊i＊i

3)③处应填()

A. if(i*i <= n)　　B. if(n%i==0)

C. while(i *i <= n)　　D. while(n%i==0)

4)④处应填

A. n >1　　B. n<=1　　C. i+i<= n　　D. i<n/i

5)⑤处应填()

A. 2　　B. i　　C. n/i　　D. n

思路解析:分解质因数原始的做法如下。本题是原始做法的一个优化版本。

```
08      for(i = ①;② <= n; i ++) {
09          ③{
10              printf("%d ", i);
11              n = n / i;
12          }
13      }
```

在原始做法里，只要发现 n 的因数，这个因数就一定是个质数(前提是每发现一个因数就要把它在 n 里除尽)因为任何一个合数都是由几个比它更小的质数组成的而这些质数一定已经在 n 里除去了。因此剩余的 n 不可能整除一些质数的乘积(即合数)。因此合数=>一定不能被整除。这句话等同于:如果能被整除=>一定不是合数(即一定是质数)

1)①处应填()

A. n-1　　B. 0　　C. 1　　D. 2

正确答案：C

解析:①处填的是第一个质数，第一个质数从 2 开始。

2)②处应填()

A. n/i　B. n/(i*i)　　C. i*i　D. i * i * i

正确答案：C

解析:枚举到√n 即可

3)③处应填()

A. if(i*i <= n)　　B. if(n%i==0)

C. while(i *i <= n)　　D. while(n%i==0)

正确答案：C

解析:要把所有的 i 都从 n 里除尽，因此需要循环。

4)④处应填

A. n >1　　B. n<=1　　C. i+i<= n　　D. i<n/i

正确答案：A

解析:根据以上思路解析的提示，如果最后 n 没有被分解成 1，说明还有一个> √n 的质因数。

5)⑤处应填()

A. 2　　B. i　　C. n/i　　D. n

正确答案：C

解析：最后剩余的 n 就是最后一个质因数。

2.(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(0 <=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。

```
01 #include <iostream>
02
03 using namespace std;
04
05 const int MAXN = 5000;
06 int n, m;
07 struct segment { int a, b; } A[MAXN];
08
09 void sort() // 排序
10 {
11   for (int i = 0; i < n; i++)
12     for (int j = 1; j < n; j++)
13       if (①)
14       {
15         segment t = A[j];
16         ②
17       }
18 }
```

```
20 int main()
21 {
22   cin >> n >> m;
23   for (int i = 0; i < n; i++)
24     cin >> A[i].a >> A[i].b;
25   sort();
26   int p = 1;
27   for (int i = 1; i < n; i++)
28     if (③)
29       A[p++] = A[i];
30   n = p;
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 }
```

1)①处应填()

A. A[j].b<A[j-1].b B.A[j].a<A[j - 1].a

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

2)②处应填()

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()

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

4)④处应填()

A　q + 1 < n & &　A [q + 1] . a < =　r

B　q + 1 < n & & A [q ＋ 1] . b < =　r

C.　q <n && A[q　+ 1].a<= r

D.　q <n&& A[q].b<= r

5)⑤处应填()

A .r = max(r,A[q + 1].a)

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

C.r = max(r，A[q].b)

D. q ＋＋

程序解析：

排序后，初步筛选一批线段。因为题目要求全覆盖所以【选择的线段的结尾】要大于【上一次选择的线段的结尾。4 号:选择。1 号:选择。6 号:不选(因为它的结尾小于【上次选择的 1 号的结尾】，所以选了很多余)。2 号:选择。3 号:选择。5 号:选择。

初步筛完线段后，还要进一步筛选。已经选出的 4，1，2，3，5 其实还是冗余的。因为最终答案是 4，2，3，5 并没有 1。那 1 有什么特点呢?可以看到 1 和 2 都属于开头比 4 的结尾，要小的，但是因为 2 的结尾更靠后就为我们实现“选择最少的线段”这个目标提供了更大的可能性。因此不选择 1。

1)①处应填()

A. A[j].b<A[j-1].b　B.A[j].a<A[j - 1].a

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

正确答案：B

解析:根据思路解析，要按起点从小到大排序即如果右边的起点小于左边的起点，那么交换。

2)②处应填()

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;

正确答案： C

解析:交换是跟前面的元素交换

3)③处应填()

A.A[i].b<A[p-1].b　B.A[i].b>A[p -1].b

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

正确答案：A

解析:根据思路解析，第一次筛选要选择终点大于【上一次选择的线段的终点】的线段，才能不多余。

3)③处应填()

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

正确答案：A

解析:根据思路解析，第一次筛选要选择终点大于【上一次选择的线段的终点】的线段，才能不多余。

4)④处应填()

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

B q + 1 < n & & A [q ＋ 1] . b < = r

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

D. q <n&& A[q].b<= r

正确答案： A

解析:根据思路解析，第二次筛选要选择所有起点小于【上一次选择的线段的终点】的线段中，终点最大的那个，就可以选尽可能少的线段。

5)⑤处应填()

A .r = max(r,A[q + 1].a)

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

C.r = max(r， A[q].b)

D. q ＋＋

正确答案： C

解析:根据思路解析，第二次筛选要选择所有起点小于【上一次选择的线段的终点】的线段中，终点最大的那个，就可以选尽可能少的线段。