CSP-J 初赛模拟卷（第 01 套）

入门级 C++ 语言试题

一、单项选择题（共 **15** 题，每题 **2** 分，共计 **30** 分）

1．中国的国家顶级域名是（ ）。

A．.cn　　B．.ch　　C．.chn　　D．.china

2．下列不属于面向对象程序设计语言的是（ ）。

A．C　　B．C++　　C．Java　　D．C#

3．以比较作为基本运算，在N个数中找出最大数，最坏情况下所需要的最少的比较次数为（ ）。

A．N^2　　B．N　　C．N-1　　D．N+1

4．表达式a*(b+c)*d 的后缀表达式为（ ），其中“*”和“+”是运算符。

A．**a+bcd　　B．abc+*d*　　C．abc+d**　　D．*a*+bcd

5．在 8 位二进制补码中，10101011 表示的数是十进制下的（ ）。

A．43　　B．-85　　C．-43　　D．-84

6．FTP 可以用于（ ）。

A．远程传输文件　　B．发送电子邮件　　C．浏览网页　　D．网上聊天

7．319和377的最大公约数是（ ）。

A．27　　B．33　　C．29　　D．31

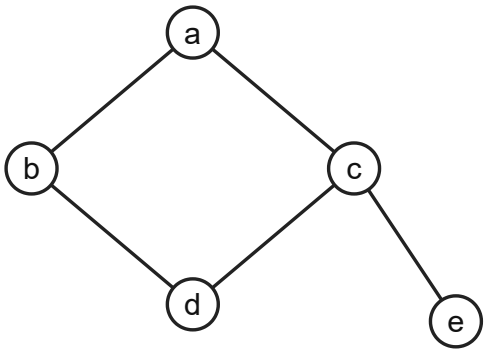
8．对于有n个顶点、m条边的无向联通图（m>n），需要删掉（ ）条边才能使其成为一棵树。

A．n-1　　B．m - n　　C．m - n - 1　　D．m - n + 1

9．如果一棵二叉树的中序遍历是 BAC，那么它的先序遍历不可能是（ ）。

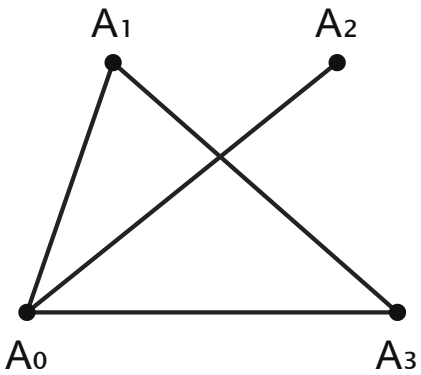
A．ABC　　B．CBA　　C．ACB　　D．BAC

10. 以a为起点，对右边的无向图进行深度优先遍历，则b,c,d,e四个点中有可能作为最后一个遍历到的点个数为（ ）。



A. 1　　B. 2　　C. 3　　D. 4

11. 以 A_0作为起点，对下面的无向图进行深度优先遍历时，遍历顺序不可能是（ ）。



A. A_0, A_1, A_2, A_3　　B. A_0, A_1, A_3, A_2　　C. A_0, A_2, A_1, A_3　　D. A_0, A_3, A_1, A_2

12. 6个人，两个人组一队，总共组成三只不区分队伍的编号，不同的组队情况有（ ）种。

A. 10　　B. 15　　C. 30　　D. 20

13. 如果根的高度为 1,具有 61 个结点的完全二叉树的高度为（ ）。

A. 5　　B. 6　　C. 7　　D. 8

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

A. $(y \vee z) \wedge x \wedge z$　　B. $x \wedge (z \vee y) \wedge z$　　C. $(x \wedge y) \wedge z$　　D. $(x \wedge y) \vee (z \vee x)$

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

二、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 ×；共 18 小题，共计 36 分，各小题分值见题目）

(1)

```
01 #include <cstdio>
02 #include <cstring>
03 using namespace std;
04 char st[100001];
05 int main() {
06     scanf("%s", st);
07     int n = strlen(st);
08     for (int i = 1; i <= n; ++i) {
09         if (n % i == 0) {
10             char c = st[i - 1];
11             if (c >= 'a')
12                 st[i - 1] = c - 'a' + 'A';
13         }
14     }
15     printf("%s", st);
16     return 0;
17 }
```

判断题

16. (1.5分) 输入的字符串只能由小写字母或大写字母组成。（ ）

17. (1.5分) 若将第 8 行的 i = 1 改为 i = 0，程序运行时会发生错误。（ ）

18. (1.5分) 若将第 8 行的 i <= n 改为 i * i <= n，程序运行结果不会改变。（ ）

19. (1.5分) 若输入的字符串全部由大写字母组成，那么输出的字符串就跟输入的字符串一样。（ ）

单选题

20. (3分) 若输入的字符串长度为 18，那么输入的字符串跟输出的字符串相比，至多有（ ）个字符不同。

A. 18　　B. 6　　C. 10　　D. 1

21. (3分) 若输入的字符串长度为（ ），那么输入的字符串跟输出的字符串相比，至多有 36 个字符不同。

A. 36　　B. 100000　　C. 1　　D. 128

(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 是不超过 10000 的正整数，完成下面的判断题和单选题。

判断题

22. (1.5分) 若 $k=1$，则输出 ans 时，len = n。（ ）

23. (1.5分) 若 $k>1$，则输出 ans 时，len 一定小于 n。（ ）

24. (1.5分) 若 $k>1$，则输出 ans 时，k^{len} 一定大于 n。（ ）

单选题

25. (3分) 若输入的 $n=10^{15}$，$k=1$，则输出等于（ ）。

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

26. (3分) 若输入的 $n=205891132094649$（即 3^{30}），$k=3$，则输出等于（ ）。

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

27. (3分) 若输入的 $n=100010002000090$，$k=10$，则输出等于（ ）。

A. 11112222444543　　B. 11122222444453

C. 11122222444543　　D. 11112222444453

(3)

```
#include <stdio.h>

int n;
int a[1000];

int f(int x)
{
    int ret = 0;
    for (; x; x &= x - 1) ret++;
    return ret;
}

int g(int x)
```

```
{
    return x & -x;
}

int main()
{
    scanf("%d", &n);
    for (int i = 0; i < n; i++) scanf("%d", &a[i]);
    for (int i = 0; i < n; i++)
        printf("%d ", f(a[i]) + g(a[i]));
    printf("\n");
    return 0;
}
```

判断题

28. (1.5分) 输入的n等于1001时，程序不会发生下标越界。（ ）

29. (1.5分) 输入的 $a[i]$ 必须全为正整数，否则程序将陷入死循环。（ ）

30. (1.5分) 当输入为“5 2 11 9 16 10”时，输出为“3 4 3 17 5”。（ ）

31. (1.5分) 当输入为”1 511998”时，输出为”18”。（ ）

32. (1.5分) 将源代码中 g 函数的定义(13-16行)移到main函数的后面，程序可以正常编译运行。（ ）

单选题

33. (3分) 当输入为”2 -65536 2147483647”时，输出为（ ）

A. “65532 33”　　B. “65552 32”　　C. “65535 34”　　D. “65554 33”

三、完善程序（程序输入不超过数组或字符串定义的范围；共 **10** 小题，共计 **30** 分，各小题分值见题目）

(1)

（最小区间覆盖）给出 n 个区间，第 i 个区间的左右端点是 $[a_i, b_i]$。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 $0 \leq i < m$ 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 （$1 \leq n \leq 5000, 1 \leq m \leq 10^9$）接下来 n 行，每行两个整数 a_i, b_i （$0 \leq a_i, b_i \leq m$）。

提示:采用贪心法解决这个问题。采用 $O(n^2)$ 的时间复杂度排序，然后贪心选择这些区间。试补全程序。

```
#include <iostream>

using namespace std;

const int MAXN = 5000;
int n, m;
struct segment { int a, b; } A[MAXN];

void sort() // 排序
{
    for (int i = 0; i < n; i++)
        for (int j = 1; j < n; j++)
            if (__(1)__)
            {
                segment t = A[j];
                __(2)__
            }
}

int main()
{
    cin >> n >> m;
    for (int i = 0; i < n; i++)
        cin >> A[i].a >> A[i].b;
    sort();
    int p = 1;
    for (int i = 1; i < n; i++)
        if (__(3)__)
            A[p++] = A[i];
    n = p;
    int ans =0, r = 0;
    int q = 0;
    while (r < m)
    {
        while (__(4)__)
            q++;
        __(5)__;
        ans++;
```

```
    }
    cout << ans << endl;
    return 0;
}
```

34. (3分) (1)处应填

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

35. (3分) (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;

36. (3分) (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

37. (3分) (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].a<=r D. q<n&&A[q].b<=r

38. (3分) (5)处应填

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)

（计数排序）计数排序是一个广泛使用的排序方法。下面的程序使用双关键字计数排序，将 n 对10000以内的整数，从小到大排序。例如有三对整数(3,4)、(2,4)、(3,3)那么排序之后应该是(2,4)、(3,3)、(3,4)。输入第一行为 n ，接下来 n 行，第 i 行有两个数 $a[i]$ 和 $b[i]$，分别表示第 i 对整数的第一关键字和第二关键字。从小到大排序后输出。
数据范围 $1 < n < 10^7, 1 < a[i], b[i] < 10^4$ 。提示:应先对第二关键字排序，再对第一关键字

排序。数组 ord[] 存储第二关键字排序的结果，数组 res[] 存储双关键字排序的结果。试补全程序。

```
01 #include <cstdio>
02 #include <cstring>
03 using namespace std;
04 const int maxn = 10000000;
05 const int maxs = 10000;
06
07 int n;
08 unsigned a[maxn], b[maxn],res[maxn], ord[maxn];
09 unsigned cnt[maxs + 1];
10 int main() {
11     scanf("%d", &n);
12     for (int i = 0; i < n; ++i)
13         scanf("%u%u", &a[i], &b[i]);
14     memset(cnt, 0, sizeof(cnt));
15     for (int i = 0; i < n; ++i)
16         __(1)__; // 利用 cnt 数组统计数量
17     for (int i = 0; i < maxs; ++i)
18         cnt[i + 1] += cnt[i];
19     for (int i = 0; i < n; ++i)
20         __(2)__; // 记录初步排序结果
21     memset(cnt, 0, sizeof(cnt));
22     for (int i = 0; i < n; ++i)
23         __(3)__; // 利用 cnt 数组统计数量
24     for (int i = 0; i < maxs; ++i)
25         cnt[i + 1] += cnt[i];
26     for (int i = n - 1; i >= 0; --i)
27         __(4)__; // 记录最终排序结果
28     for (int i = 0; i < n; i++)
29         printf("%u %u\n", __(5)__);
30
31     return 0;
32 }
```

39. (3分) (1)处应填

A. ++cnt [i]　　B. ++cnt[b[i]]

C. ++cnt[a[i] * maxs + b[i]]　　D. ++cnt[a[i]]

40. (3分) (2)处应填

A. ord[--cnt[a[i]]] = i　　B. ord[--cnt[b[i]]] = a[i]
C. ord[--cnt[a[i]]] = b[i]　D. ord[--cnt[b[i]]] = i

41. (3分) (3)处应填

A. ++cnt[b[i]]　B. ++cnt[a[i] * maxs + b[i]]
C. ++cnt[a[i]]　D. ++cnt [i]

42. (3分) (4)处应填

A. res[--cnt[a[ord[i]]]] = ord[i]　B. res[--cnt[b[ord[i]]]] = ord[i]
C. res[--cnt[b[i]]] = ord[i]　　D. res[--cnt[a[i]]] = ord[i]

43. (3分) (5)处应填

A. a[i], b[i]　　B. a[res[i]], b[res[i]]
C. a[ord[res[i]], b[ord[res[i]]　D. a[res[ord[i]], b[res[ord[i]])