CSP-J 初赛模拟卷（第 02 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．以下哪一种设备属于输出设备：（ ）

A．扫描仪　　B．键盘　　C．鼠标　　D．打印机

2．如果256 种颜色用二进制编码来表示，至少需要（ ）位。

A．6　　B．7　　C．8　　D．9

3．对于入栈顺序为a,b,c,d,e的序列，下列（ ）不是合法的出栈序列。

A．a,b,c,d,e　　B．e,d,c,b,a　　C．b,a,c,d,e　　D．c,d,a,e,b

4．如果一棵二叉树只有根结点，那么这棵二叉树高度为1。请问高度为5的完全二叉树有（ ）种不同形态?

A．16　　B．15　　C．17　　D．32

5．二进制数 11.01 在十进制下是（ ）。

A．3.25　　B．4.125　　C．6.25　　D．11.125

6．下列各无符号十进制整数中，能用八位二进制表示的数中最大的是（ ）。

A．296　　B．133　　C．256　　D．199

7．一些数字可以颠倒过来看，例如0、1、8颠倒过来还是本身，6颠倒过来是9，9颠倒过来看还是6,其他数字颠倒过来都不构成数字。 类似的，一些多位数也可以颠倒过来看，比如106颠倒过来是901。假设某个城市的车牌只由5位数字组成，每一位都可以取0到9。 请问这个城市最多有多少个车牌倒过来恰好还是原来的车牌？（ ）

A．60　　B．125　　C．75　　D．100

8．二进制数101.11对应的十进制数是（ ）。

A．6.5　　B．5.5　　C．5.75　　D．5.25

9．使用冒泡排序对序列进行升序排列，每执行一次交换操作系统将会减少 1 个逆序对，因此序列 5，4，3，2，1 需要执行（ ）次操作，才能完成冒泡排序。

A．0　　B．5　　C．10　　D．15

10. 由1, 1, 2, 2, 3这五个数字组成不同的三位数有（ ）种。

A. 18　　B. 15　　C. 12　　D. 24

11. 将（2,6,10,17）分别存储到某个地址区间为0~10的哈希表中，如果哈希函数 $h(x)=$（ ），将不会产生冲突，其中a mod b表示a除以b的余数。

A. x mod 11　　B. $x^2 \mod 11$

C. (2x) mod 11　　D. $\lfloor sqrt(x) \rfloor \mod 11$，其中 $\lfloor sqrt(x) \rfloor$ 表示 $sqrt(x)$ 下取整

12. 寄存器是（ ）的重要组成部分.

A. 硬盘　　B. 高速缓存　　C. 内存　　D. 中央处理器（CPU）

13. 某算法的时间复杂度满足递推式 $T(n)=T(n-1)+n$，边界条件为 $T(0)=1$，则该算法的时间复杂度为（ ）。

A. $O(\log n)$　　B. $O(n\log n)$　　C. $O(n)$　　D. $O(n^2)$

14. 100以内最大的素数是（ ）。

A. 89　　B. 97　　C. 91　　D. 93

15. 言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:
    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 个数用以上言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

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 **40.5** 分，各小题分值见题目）

(1)

```
01 #include <stdio.h>
02 #include <string.h>
03
04 char base[64];
05 char table[256];
06 char str[256];
07 char ans[256];
08
09 void init()
10 {
11     for (int i = 0; i < 26; i++) base[i] = 'A' + i;
12     for (int i = 0; i < 26; i++) base[26 + i] = 'a' + i;
13     for (int i = 0; i < 10; i++) base[52 + i] = '0' + i;
14     base[62] = '+', base[63] = '/';
15
16     for (int i = 0; i < 256; i++) table[i] = 0xff;
17     for (int i = 0; i < 64; i++) table[base[i]] = i;
18     table['='] = 0;
19 }
20
21 void decode(char *str)
22 {
23     char *ret = ans;
24     int i, len = strlen(str);
25     for (i = 0; i < len; i += 4) {
26         (*ret++) = table[str[i]] << 2 | table[str[i + 1]] >> 4;
27         if (str[i + 2] != '=')
28             (*ret++) = (table[str[i + 1]] & 0x0f) << 4 | table[str[i
    + 2]] >> 2;
29         if(str[i + 3] != '=')
30             (*ret++) = table[str[i + 2]] << 6 | table[str[i + 3]];
31     }
32 }
33
34 int main()
35 {
36     init();
37     printf("%d\n", (int)table[0]);
```

```
38
39      scanf("%s", str);
40      decode(str);
41      printf("%s\n", ans);
42      return 0;
43  }
```

假设 char 是 8 位有符号类型，输入字符串长度为 4 的倍数，且输入不超过数组范围。

判断题

16. (1.5分) 输出的第二行一定是由小写字母、大写字母、数字和 +、/、= 构成的字符串。（ ）

17. (1.5分) 可能存在输入不同，但输出的第二行相同的情形。（ ）

18. (1.5分) 输出的第一行为 -1。（ ）

单选题

19. (3分) 设输入字符串长度为 n，decode 函数的时间复杂度为（ ）。

A. $O(\sqrt{n})$　　B. $O(n)$　　C. $O(n\log n)$　　D. $O(n^2)$

20. (3分) 当输入为 Y3Nx 时，输出的第二行为（ ）。

A. csp　　B. csq　　C. CSP　　D. Csp

21. (3.5分) 当输入为 Y2NmIDIwMjE= 时，输出的第二行为（ ）。

A. ccf2021　　B. ccf2022　　C. ccf 2021　　D. ccf 2022

(2)

```
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，$d[i][0]$、$d[i][i]$ 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

22. (1.5分) 若输入 n 为0，此程序可能会死循环或发生运行错误。（ ）

23. (1.5分) 若输入 n 为20,接下来的输入全为0，则输出为0。（ ）

24. (1.5分) 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

25. (3分) 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（ ）。

A. 1890　　B. 1881　　C. 1908　　D. 1917

26. (3分) 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（ ）。

A. 2000　　B. 2010　　C. 2030　　D. 2020

27. (4分) 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. 2440　　B. 2220　　C. 2240　　D. 2420

(3)

```
01 #include<cstdio>
02 using namespace std;
03 int n, m;
04 int a[100], b[100];
05
06 int main() {
07     scanf("%d%d", &n, &m);
08     for (int i = 1; i <= n; ++i)
09         a[i] = b[i] = 0;
10     for (int i = 1; i <= m; ++i) {
11         int x, y;
12         scanf("%d%d", &x, &y);
13         if (a[x] < y && b[y] < x) {
14             if (a[x] > 0)
15                 b[a[x]] = 0;
16             if (b[y] > 0)
17                 a[b[y]] = 0;
18             a[x] = y;
19             b[y] = x;
20         }
21     }
22     int ans = 0;
23     for (int i = 1; i <= n; ++i) {
24         if (a[i] == 0)
25             ++ans;
26         if (b[i] == 0)
27             ++ans;
```

```
    }
    printf("%d", ans);
    return 0;
}
```

假设输入的 n 和 m 都是正整数，x 和 y 都是在 $[1,n]$ 的范围内的整数，完成下面的判断题和单选题：

判断题

28. (1.5分) 当 $m>0$ 时，输出的值一定小于 $2n$。（ ）

29. (1.5分) 执行完第27行的“+ + ans”时，ans一定是偶数。（ ）

30. (1.5分) $a[i]$ 和 $b[i]$ 不可能同时大于0。（ ）

31. (1.5分) 右程序执行到第13行时， x 总是小于 y ，那么第15行不会被执行。（ ）

单选题

32. (3分) 若 m 个 x 两两不同，且 m 个 y 两两不同，则输出的值为（ ）

A. $2n-2m$　　B. $2n+2$　　C. $2n-2$　　D. $2n$

33. (3分) 若 m 个 x 两两不同，且 m 个 y 都相等，则输出的值为（ ）

A. $2n-2$　　B. $2n$　　C. $2m$　　D. $2n-2m$

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)

（矩阵变幻）有一个奇幻的矩阵，在不停的变幻，其变幻方式为:数字0变成矩阵 0 0 0 1

数字1变成矩阵 1 1 1 0

最初该矩阵只有一个元素 0，变幻 n 次后，矩阵会变成什么样？

例如，矩阵最初为:[0]； 矩阵变幻 1 次后: 0 0 0 1

矩阵变幻 2 次后: 0 0 0 0 0 1 0 1 0 0 1 1 0 1 1 0

输入一行一个不超过10的正整数n。输出变幻n次后的矩阵。

试补全程序。

提示：

<< 表示二进制左移运算符，例如 $(11)_2 << 2 = (1100)_2$ ；而^表示二进制异或运算符，它将两个参与运算的数中的每个对应的二进制位进行比较，若两个二进制位相同，则运算结果的对应二进制位为0，反之为1。

```
#include <cstdio>
using namespace std;
int n;
const int max_size = 1 << 10;

int res[max_size][max_size];

void recursive(int x, int y, int n, int t) {
    if (n == 0) {
        res[x][y] = __(1)__;
        return;
    }
    int step = 1 << (n - 1);
    recursive(__(2)__, n - 1, t);
    recursive(x, y + step, n - 1, t);
    recursive(x + step, y, n - 1, t);
    recursive(__(3)__, n - 1, !t);
}

int main() {
    scanf("%d", &n);
    recursive(0, 0, __(4)__);
    int size = __(5)__;
    for (int i = 0; i < size; i++) {
        for (int j = 0; j < size; j++)
            printf("%d", res[i][j]);
        puts("");
    }
    return 0;
}
```

34. (3分) (1)处应填

A. n%2　　B. 0　　C. t　　D. 1

35. (3分) (2)处应填

A. x-step,y-step　B. x,y-step　C. x-step,y　D. x,y

36. (3分) (3)处应填

A. x-step,y-step　B. x+step,y+step　C. x-step,y　D. x,y-step

37. (3分) (4)处应填

A. n-1,n%2　　B. n, 0　　C. n,n%2　　D. n-1,0

38. (3分) (5)处应填

A. $1 \ll (n+1)$　　B. 1<<n　　C. $n+1$　　D. 1<<(n-1)

(2)

（矩形计数）平面上有*n*个关键点，求有多少个四条边都和*x*轴或者*y*轴平行的矩形，满足四个顶点都是关键点。给出的关键点可能有重复，但完全重合的矩形只计一次。 试补全枚举算法。

```
01 #include <stdio.h>
02
03 struct point {
04     int x, y, id;
05 };
06
07 int equals(struct point a, struct point b){
08     return a.x == b.x && a.y == b.y;
09 }
10
11 int cmp(struct point a, struct point b){
12     return  __(1)__;
13 }
14
15 void sort(struct point A[], int n){
16     for (int i = 0; i < n; i++)
17         for (int j = 1; j < n; j++)
18             if (cmp(A[j], A[j - 1])){
19                 struct point t = A[j];
20                 A[j] = A[j - 1];
21                 A[j - 1] = t;
22             }
23 }
24
25 int unique(struct point A[], int n){
26     int t = 0;
27     for (int i = 0; i < n; i++)
```

```
28          if (__(2)__)
29              A[t++] = A[i];
30      return t;
31  }
32
33  int binary_search(struct point A[], int n, int x, int y){
34      struct point p;
35      p.x = x;
36      p.y = y;
37      p.id = n;
38      int a = 0, b = n - 1;
39      while(a < b){
40          int mid = __(3)__;
41          if (__(4)__)
42              a = mid + 1;
43          else
44              b = mid;
45      }
46      return equals(A[a], p);
47  }
48
49  #define MAXN 1000
50  struct point A[MAXN];
51
52  int main(){
53      int n;
54      scanf("%d", &n);
55      for (int i = 0; i < n; i++){
56          scanf("%d %d", &A[i].x, &A[i].y);
57          A[i].id = i;
58      }
59      sort(A, n);
60      n = unique(A, n);
61      int ans = 0;
62      for (int i = 0; i < n; i++)
63          for (int j = 0; j < n; j++)
64              if ( __(5)__ && binary_search(A, n, A[i].x, A[j].y) &&
    binary_search(A, n, A[j].x, A[i].y)){
65                  ans++;
66              }
67      printf("%d\n", ans);
68      return 0;
69  }
```

39. (3分) (1)处应填

A. a.x != b.x ? a.x < b.x : a.id < b.id
B. a.x != b.x ? a.x < b.x : a.y < b.y
C. equals(a,b) ? a.id < b.id : a.x < b.x
D. equals(a,b) ? a.id < b.id : (a.x != b.x ? a.x < b.x : a.y < b.y)

40. (3分) (2)处应填

A. i == 0 || cmp(A[i], A[i-1])　　B. t == 0 || equals(A[i], A[t-1])
C. i == 0 || !cmp(A[i], A[i-1])　　D. t == 0 || !equals(A[i], A[t-1])

41. (3分) (3)处应填

A. b - (b-a)/2 + 1　　B. (a+b+1) >> 1
C. (a+b) >> 1　　D. a + (b-a+1)/2

42. (3分) (4)处应填

A. !cmp(A[mid], p)　　B. cmp(A[mid], p)
C. cmp(p, A[mid])　　D. !cmp(p, A[mid])

43. (3分) (5)处应填

A. A[i].x == A[j].x
B. A[i].id < A[j].id
C. A[i].x == A[j].x && A[i].id < A[j].id
D. A[i].x < A[j].x && A[i].y < A[j].y