CSP-J 初赛模拟卷（第 14 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）

1．一个 32 位整型变量占用（ ）个字节。

A．4　　B．8　　C．32　　D．128

2．二进制数 00100100 和 00010101 的和是（ ）。

A．00101000　　B．001010100　　C．01000101　　D．00111001

3．关于汇编语言，下列说法错误的是（ ）。

A．是一种与具体硬件相关的程序设计语言
B．在编写复杂程序时，相对于高级语言而言代码量较大，且不易调试
C．可以直接访问寄存器、内存单元、以及I/O端口
D．随着高级语言的诞生，如今已完全被淘汰，不再使用

4．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

5．原字符串中任意一段连续的字符所组成的新字符串称为子串。 则字符“AAABBBCCC”共有（ ）个不同的非空子串。

A．3　　B．12　　C．36　　D．45

6．设变量x为非负float型且已赋值，则以下语句中能将x中的数值保留到小数点后两位，并将第三位四舍五入的是（ ）。

A．x = (x * 100) + 0.5 / 100.0;　　B．x = (x * 100 + 0.5) / 100.0;
C．x = (int)(x * 100 + 0.5)/100.0;　　D．x = (x / 100 + 0.5) * 100.0;

7．一个字长为8位的整数的补码是11111001，则它的原码是（ ）。

A．00000111　　B．01111001　　C．11111001　　D．10000111

8．下面是根据欧几里得算法编写的函数，它所计算的是a和b的（ ）。 int euclid(int a, int b) { if (b == 0) return a; else return euclid(b, a % b); }

A．最大公共质因子　　B．最小公共质因子　　C．最大公约数　　D．最小公倍数

9．有7个一模一样的苹果，放到3个一样的盘子中，盘子可以为空，一共有（ ）种放法。

A．7　　B．8　　C．21　　D．3^7

10. 如果树根算第1层，那么一棵 n 层的二叉树最多有（ ）个结点。

A. 2^n-1　　B. 2^n　　C. 2^n+1　　D. 2^{n+1}

11. 无向完全图是图中每对顶点之间都恰好有一条边的简单图。已知无向完全图G有7个顶点，则它共有（ ）条边。

A. 7　　B. 21　　C. 42　　D. 49

12. 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

13. 对于给定的序列 $\{a_k\}$，我们把 (i,j) 称为逆序对当且仅当 $i<j$ 且 $a_i>a_j$。那么序列 $\{1,7,2,3,5,4\}$ 的逆序对数为（ ）个。

A. 4　　B. 5　　C. 6　　D. 7

14. 给定含有 n 个互不相同元素的单峰数组 L，满足先严格递增、后严格递减，峰顶位于数组内部。用下面的二分算法寻找峰顶，请将 a、b、c 三行代码依次填入空白处。正确的填空顺序是（ ）。

```
a. return Search(k + 1, r)
b. return Search(l, k - 1)
c. return L[k]

Search(l, r):
    if l == r then return L[l]
    k = floor((l + r) / 2)
    if L[k] > L[k - 1] and L[k] > L[k + 1]: ______
    else if L[k] < L[k + 1]: ______
    else: ______

调用 Search(2, n - 1)
```

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

15. 广度优先搜索时，需要用到的数据结构是（ ）。

A. 链表　　B. 队列　　C. 栈　　D. 散列表

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 **37** 分，各小题分值见题目）

(1)

```
01 #include<iostream>
02 using namespace std;
03 #define MAXN 20
04 int gu[MAXN][MAXN];
05 int luo(int n, int m) {
06     if(n <= 1 || m < 2)
07         return 1;
08     if(gu[n][m] != -1)
09         return gu[n][m];
10     int ans=0;
11     for(int i = 0; i <m; i += 2)
12         ans+=luo(n-1, i);
13     gu[n][m] = ans;
14     return ans;
15 }
16 int main() {
17     int n, m;
18     cin >> n >> m;
19     for(int i = 0; i < MAXN; i++)
20         for(int j = 0; j < MAXN; j++)
21             gu[i][j]= -1;
22     cout << luo(n, m);
23     return 0;
24 }
```

判断题

16. (1分) luo函数中，m的值不可能是奇数。（ ）

17. (1分) 若将第11行的”<“改为”<=“,程序的输出结果可能会改变。（ ）

18. (2分) 若将第8、9、13行删除，程序的运行的结果不变。（ ）

19. (2分) 在添加合适的头文件后，将第19到21行替换为“memset(gu, 255,sizeof(gu))；”可以起到相同的作用。（ ）

单选题

20. (4分) 若输入数据为4 8，则输出为（ ）。

A. 7　　B. 8　　C. 15　　D. 16

21. (3分) 最坏情况下，此程序的时间复杂度是（ ）。

A. O(m²n)　　B. O(nm!)　　C. O(n²)　　D. O(n²m)

(2)

```
01 #include <iostream>
02 using namespace std;
03
04 long long n, ans;
05 int k, len;
06 long long d[1000000];
07
08 int main() {
09     cin >> n >> k;
10     d[0] = 0;
11     len= 1;
12     ans = 0;
13     for (long long i = 0; i <n; ++i) {
14         ++d[0];
15         for (int j = 0; j + 1<len; ++j) {
16             if (d[j] == k) {
17                 d[j] = 0;
18                 d[j + 1] += 1;
19                 ++ans;
20             }
21         }
22         if (d[len - 1] == k) {
23             d[len - 1] = 0;
24             d[len] =1;
25             ++len;
26             ++ans;
27         }
28     }
29     cout << ans << endl;
30     return 0;
31 }
```

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

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. (1.5分) 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}

26. (3分) 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. $(3^{30}+1)/2$

27. (3分) 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

(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;
28      }
29      printf("%d", ans);
30      return 0;
31  }
```

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 **29** 分，各小题分值见题目）

(1)

完善程序：

（打印日历）输入月份 $m(1 \leq m \leq 12)$，按一定格式打印2015年第 m 月的月历。例如，2015年1月的月历打印效果如下(第一列为周日)：

S	M	T	W	T	F	S
				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	26	27	28	29	30	31

```
01 #include <iostream>
02 #include <string>
03 using namespace std;
04 const int dayNum[] = {-1, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31,
   30, 31};
05 int m, offset, i;
06 int main()
07 {
08     cin >> m;
09     cout << "S\tM\tT\tW\tT\tF\tS" << endl; /* '\t'为 TAB 制表符 */
10     __(1)__;
11     for (i = 1; i < m; i++)
12         offset = __(2)__;
13     for (i = 0; i < offset; i++)
14         cout << '\t';
15     for (i = 1; i <= __(3)__; i++)
16     {
17         cout << __(4)__;
18         if (i == dayNum[m] || __(5)__ == 0)
19             cout << endl;
20         else
21             cout << '\t';
22     }
23     return (0);
24 }
```

34. (3分) (1)处应填：_____

35. (3分) (2)处应填：_____

36. (2.5分) (3)处应填：_____

37. (2.5分) (4)处应填：_____

38. (3分) (5)处应填：_____

(2)

（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。

```
01 #include <cstdio>
02 using namespace std;
03 int n;
04 const int max_size = 1 << 10;
05
06 int res[max_size][max_size];
07
08 void recursive(int x, int y, int n, int t) {
09     if (n == 0) {
10         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;
}
```

39. (3分) (1)处应填

A. n%2　　B. 0　　C. t　　D. 1

40. (3分) (2)处应填

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

41. (3分) (3)处应填

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

42. (3分) (4)处应填

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

43. (3分) (5)处应填

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