第二十三届全国青少年信息学奥林匹克联赛初赛

提高组 C++语言试题

竞赛时间：2017 年 10 月 14 日 14:30~16:30

选手注意：

- 试题纸共有 10 页，答题纸共有 2 页，满分 100 分。请在答题纸上作答，写在试题纸上的一律无效。
- 不得使用任何电子设备（如计算器、手机、电子词典等）或查阅任何书籍资料。

一、单项选择题（共 15 题，每题 1.5 分，共计 22.5 分；每题有且仅有一个正确选项）

1. 从（ ）年开始，NOIP 竞赛将不再支持 Pascal 语言。
   A. 2020　　B. 2021　　C. 2022　　D. 2023

2. 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

3. 分辨率为 1600x900、16 位色的位图，存储图像信息所需的空间为（ ）。
   A. 2812.5KB　　B. 4218.75KB　　C. 4320KB　　D. 2880KB

4. 2017 年 10 月 1 日是星期日，1949 年 10 月 1 日是（ ）。
   A. 星期三　　B. 星期日　　C. 星期六　　D. 星期二

5. 设 G 是有 n 个结点、m 条边（$n \le m$）的连通图，必须删去 G 的（ ）条边，才能使得 G 变成一棵树。
   A. $m-n+1$　　B. $m-n$　　C. $m+n+1$　　D. $n-m+1$

6. 若某算法的计算时间表示为递推关系式：
   $T(N) = 2T(N/2) + N \log N$
   $T(1) = 1$
   则该算法的时间复杂度为（ ）。
   A. $O(N)$　　B. $O(N \log N)$　　C. $O(N \log^2 N)$　　D. $O(N^2)$

7. 表达式 a * (b + c) * d 的后缀形式是（ ）。
   A. a b c d * + *　　B. a b c + * d *　　C. a * b c + * d　　D. b + c * a * d

8. 由四个不同的点构成的简单无向连通图的个数是（ ）。
   A. 32　　B. 35　　C. 38　　D. 41

9. 将 7 个名额分给 4 个不同的班级，允许有的班级没有名额，有（　）种不同的分配方案。

A. 60　　B. 84　　C. 96　　D. 120

10. 若 f[0] = 0, f[1] = 1, f[n + 1] = (f[n] + f[n - 1]) / 2，则随着 i 的增大，f[i]将接近于（　）。

A. 1/2　　B. 2/3　　C. $\frac{\sqrt{5}-1}{2}$　　D. 1

11. 设 A 和 B 是两个长为 n 的有序数组，现在需要将 A 和 B 合并成一个排好序的数组，请问任何以元素比较作为基本运算的归并算法最坏情况下至少要做（　）次比较。

A. n^2　　B. $n \log n$　　C. 2n　　D. 2n-1

12. 在 n（$n \geq 3$）枚硬币中有一枚质量不合格的硬币（质量过轻或质量过重），如果只有一架天平可以用来称重且称重的硬币数没有限制，下面是找出这枚不合格的硬币的算法。请把 a-c 三行代码补全到算法中。

a. $A \leftarrow X \cup Y$
b. $A \leftarrow Z$
c. $n \leftarrow |A|$

算法 Coin(A, n)
1. $k \leftarrow \lfloor n/3 \rfloor$
2. 将 A 中硬币分成 X，Y，Z 三个集合，使得$|X| = |Y| = k, |Z| = n - 2k$
3. if $W(X) \neq W(Y)$　　//W(X), W(Y)分别为 X 或 Y 的重量
4. then _____
5. else _____
6. _____
7. if n>2 then goto 1
8. if n=2 then 任取 A 中 1 枚硬币与拿走硬币比较，若不等，则它不合格；若相等，则 A 中剩下的硬币不合格.
9. if n=1 then A 中硬币不合格

正确的填空顺序是（　）。

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

13. 有正实数构成的数字三角形排列形式如图所示。第一行的数为 a_{11}；第二行的数从左到右依次为 a_{21}, a_{22}；…第 n 行的数为 a_{n1}, a_{n2}, …, a_{nn}。从 a_{11} 开始，每一行的数 a_{ij} 只有两条边可以分别通向下一行的两个数 $a_{(i+1)j}$ 和 $a_{(i+1)(j+1)}$。用动态规划算法找出一条从 a_{11} 向下通到 a_{n1}, a_{n2}, …, a_{nn} 中某

a_{11}
a_{21}　a_{22}
a_{31}　a_{32}　a_{33}
……
a_{n1}　a_{n2} ……　a_{nn}

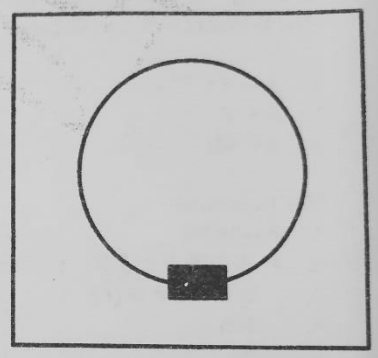
个数的路径，使得该路径上的数之和达到最大。
令 C[i,j]是从 a_{11} 到 a_{ij} 的路径上的数的最大和，并且 C[i,0]=C[0,j]=0，则 C[i,j]=（　）。

A. max{C[i-1,j-1], C[i-1,j]} + a_{ij}
B. C[i-1,j-1] + C[i-1,j]
C. max{C[i-1,j-1], C[i-1,j]} + 1
D. max{C[i,j-1],C[i-1,j]} + a_{ij}

14. 小明要去南美洲旅游，一共乘坐三趟航班才能到达目的地，其中第 1 个航班准点的概率是 0.9，第 2 个航班准点的概率为 0.8，第 3 个航班准点的概率为 0.9。如果存在第 i 个（i=1,2）航班晚点，第 i+1 个航班准点，则小明将赶不上第 i+1 个航班，旅行失败；除了这种情况，其他情况下旅行都能成功。请问小明此次旅行成功的概率是（　）。

A. 0.5　　B. 0.648　　C. 0.72　　D. 0.74

15. 欢乐喷球：儿童游乐场有个游戏叫“欢乐喷球”，正方形场地中心能不断喷出彩色乒乓球，以场地中心为圆心还有一个圆形轨道，轨道上有一列小火车在匀速运动，火车有六节车厢。假设乒乓球等概率落到正方形场地的每个地点，包括火车车厢。小朋友玩这个游戏时，只能坐在同一个火车车厢里，可以在自己的车厢里捡落在该车厢内的所有乒乓球，每个人每次游戏有三分钟时间，则一个小朋友独自玩一次游戏期望可以得到（　）个乒乓球。假设乒乓球喷出的速度为 2 个/秒，每节车厢的面积是整个场地面积的 1/20。



A. 60　　B. 108　　C. 18　　D. 20

二、不定项选择题（共 5 题，每题 1.5 分，共计 7.5 分；每题有一个或多个正确选项，多选或少选均不得分）

1. 以下排序算法在最坏情况下时间复杂度最优的有（　）。

A. 冒泡排序　　B. 快速排序　　C. 归并排序　　D. 堆排序

2. 对于入栈顺序为 a, b, c, d, e, f, g 的序列，下列（　）不可能是合法的出栈序列。

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

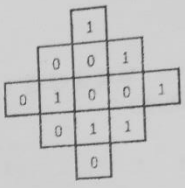
3. 下列算法中，（　）是稳定的排序算法。
A. 快速排序　　B. 堆排序　　C. 希尔排序　　D. 插入排序

4. 以下是面向对象的高级语言的有（　）。
A. 汇编语言　　B. C++　　C. Fortran　　D. Java

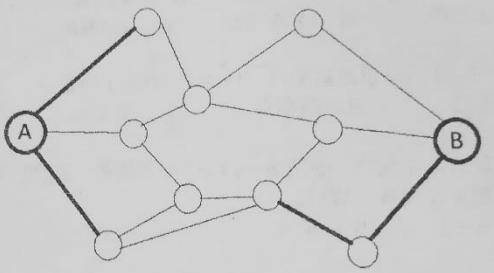
5. 以下和计算机领域密切相关的奖项有（　）。
A. 奥斯卡奖　　B. 图灵奖　　C. 诺贝尔奖　　D. 王选奖

三、问题求解（共 2 题，每题 5 分，共计 10 分）

1. 如右图所示，共有 13 个格子。对任何一个格子进行一次操作，会使得它自己以及与它上下左右相邻的格子中的数字改变（由 1 变 0，或由 0 变 1）。现在要使得所有的格子中的数字都变为 0，至少需要_____次操作。



2. 如下图所示，A 到 B 是连通的。假设删除一条细的边的代价是 1，删除一条粗的边的代价是 2，要让 A、B 不连通，最小代价是_____（2 分），最小代价的不同方案数是_____（3 分）。（只要有一条删除的边不同，就是不同的方案）



四、阅读程序写结果（共 4 题，每题 8 分，共计 32 分）

1.
```
#include <iostream>
using namespace std;

int g(int m, int n, int x) {
    int ans = 0;
    int i;
    if (n == 1)
```

```
        return 1;
    for (i = x; i <= m / n; i++)
        ans += g(m - i, n - 1, i);
    return ans;
}

int main() {
    int t, m, n;
    cin >> m >> n;
    cout << g(m, n, 0) << endl;
    return 0;
}
```

输入：8 4
输出：_____

2.
```
#include <iostream>
using namespace std;

int main() {
    int n, i, j, x, y, nx, ny;
    int a[40][40];
    for (i = 0; i < 40; i++)
        for (j = 0; j < 40; j++)
            a[i][j] = 0;
    cin >> n;
    y = 0; x = n - 1;
    n = 2 * n - 1;
    for (i = 1; i <= n * n; i++) {
        a[y][x] = i;
        ny = (y - 1 + n) % n;
        nx = (x + 1) % n;
        if ((y == 0 && x == n - 1) || a[ny][nx] != 0)
            y = y + 1;
        else { y = ny; x = nx; }
    }
    for (j = 0; j < n; j++)
        cout << a[0][j] << " ";
    cout << endl;
    return 0;
}
```

输入：3

输出：_____

3.
```
#include <iostream>
using namespace std;

int n, s, a[100005], t[100005], i;
void mergesort(int l, int r) {
    if (l == r)
        return;
    int mid = (l + r) / 2;
    int p = l;
    int i = l;
    int j = mid + 1;
    mergesort(l, mid);
    mergesort(mid + 1, r);
    while (i <= mid && j <= r) {
        if (a[j] < a[i]) {
            s += mid - i + 1;
            t[p] = a[j];
            p++;
            j++;
        }
        else {
            t[p] = a[i];
            p++;
            i++;
        }
    }
    while (i <= mid) {
        t[p] = a[i];
        p++;
        i++;
    }
    while (j <= r) {
        t[p] = a[j];
        p++;
        j++;
    }
    for (i = l; i <= r; i++)
        a[i] = t[i];
}

int main() {
```

```
    cin >> n;
    for (i = 1; i <= n; i++)
        cin >> a[i];
    mergesort(1, n);
    cout << s << endl;
    return 0;
}
```

输入：6
2 6 3 4 5 1
输出：_____

4.
```
#include <iostream>
using namespace std;

int main() {
    int n, m;
    cin >> n >> m;
    int x = 1;
    int y = 1;
    int dx = 1;
    int dy = 1;
    int cnt = 0;
    while (cnt != 2) {
        cnt = 0;
        x = x + dx;
        y = y + dy;
        if (x == 1 || x == n) {
            ++cnt;
            dx = -dx;
        }
        if (y == 1 || y == m) {
            ++cnt;
            dy = -dy;
        }
    }
    cout << x << " " << y << endl;
    return 0;
}
```

输入 1：4 3
输出 1：_____（2 分）
输入 2：2017 1014

输出 2：_____（3 分）
输入 3：987 321
输出 3：_____（3 分）

五、完善程序（共 2 题，每题 14 分，共计 28 分）

1. （大整数除法）给定两个正整数 p 和 q，其中 p 不超过 10^{100}，q 不超过 100000，求 p 除以 q 的商和余数。（第一空 2 分，其余 3 分）

输入：第一行是 p 的位数 n，第二行是正整数 p，第三行是正整数 q。
输出：两行，分别是 p 除以 q 的商和余数。

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

int p[100];
int n, i, q, rest;
char c;

int main() {
    cin >> n;
    for (i = 0; i < n; i++) {
        cin >> c;
        p[i] = c - '0';
    }
    cin >> q;
    rest = ____(1)____;
    i = 1;
    while (____(2)____ && i < n) {
        rest = rest * 10 + p[i];
        i++;
    }
    if (rest < q)
        cout << 0 << endl;
    else {
        cout << ____(3)____;
        while (i < n) {
            rest = ____(4)____;
            i++;
            cout << rest / q;
        }
        cout << endl;
    }
```

```
    cout << ____(5)____ << endl;
    return 0;
}
```

2. （**最长路径**）给定一个有向无环图，每条边长度为 1，求图中的最长路径长度。（第五空 2 分，其余 3 分）

输入：第一行是结点数 n（不超过 100）和边数 m，接下来 m 行，每行两个整数 a, b，表示从结点 a 到结点 b 有一条有向边。结点标号从 0 到(n-1)。

输出：最长路径长度。

提示：先进行拓扑排序，然后按照拓扑序计算最长路径。

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

int n, m, i, j, a, b, head, tail, ans;
int graph[100][100];    // 用邻接矩阵存储图
int degree[100];        // 记录每个结点的入度
int len[100];           // 记录以各结点为终点的最长路径长度
int queue[100];         // 存放拓扑排序结果

int main() {
    cin >> n >> m;
    for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
            graph[i][j] = 0;
    for (i = 0; i < n; i++)
        degree[i] = 0;
    for (i = 0; i < m; i++) {
        cin >> a >> b;
        graph[a][b] = 1;
        ____(1)____;
    }
    tail = 0;
    for (i = 0; i < n; i++)
        if (____(2)____) {
            queue[tail] = i;
            tail++;
        }
    head = 0;
    while (tail < n - 1) {
        for (i = 0; i < n; i++)
            if (graph[queue[head] ][i] == 1) {
                ____(3)____;
```

```
                if (degree[i] == 0) {
                    queue[tail] = i;
                    tail++;
                }
            }
            (4);
    }
    ans = 0;
    for (i = 0; i < n; i++) {
        a = queue[i];
        len[a] = 1;
        for (j = 0; j < n; j++)
            if (graph[j][a] == 1 && len[j] + 1 > len[a])
                len[a] = len[j] + 1;
        if ((5))
            ans = len[a];
    }
    cout << ans << endl;
    return 0;
}
```