第二十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：2014 年 10 月 12 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. 以下哪个是面向对象的高级语言（　　）。

A. 汇编语言　　B. C++　　C. Fortran　　D. Basic

2. 1TB 代表的字节数量是（　　）。

A. 2 的 10 次方　　B. 2 的 20 次方　　C. 2 的 30 次方　　D. 2 的 40 次方

3. 二进制数 00100100 和 00010101 的和是（　　）。

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

4. TCP 协议属于哪一层协议（　　）。

A. 应用层　　B. 传输层　　C. 网络层　　D. 数据链路层

5. 下列几个 32 位 IP 地址中，书写错误的是（　　）。

A. 162.105.142.27　　B. 192.168.0.1　　C. 256.256.129.1　　D. 10.0.0.1

6. 在无向图中，所有顶点的度数之和是边数的（　　）倍。

A. 0.5　　B. 1　　C. 2　　D. 4

7. 对长度为 n 的有序单链表，若检索每个元素的概率相等，则顺序检索到表中任一元素的平均检索长度为（　　）。

A. n/2　　B. (n+1)/2　　C. (n-1)/2　　D. n/4

8. 编译器的主要功能是（　　）。

A. 将一种高级语言翻译成另一种高级语言

B. 将源程序翻译成指令

C. 将低级语言翻译成高级语言

D. 将源程序重新组合

9. 二进制数 111.101 所对应的十进制数是（　）。

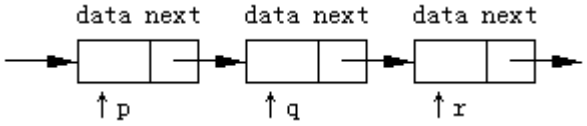
A. 5.625　　B. 5.5　　C. 6.125　　D. 7.625

10. 若有变量 `int a,float x,y`，且 `a=7,x=2.5,y=4.7`，则表达式 `x+a%3*(int)(x+y)%2/4` 的值大约是（　）。

A. 2.500000　　B. 2.750000　　C. 3.500000　　D. 0.000000

11. 有以下结构体说明和变量定义，如图所示，指针 p、q、r 分别指向一个链表中的三个连续结点。

```
struct node {
    int data;
    node *next;
} *p, *q, *r;
```



现要将 `q` 和 `r` 所指结点的先后位置交换，同时要保持链表的连续，以下程序段中错误的是（　）。

A. `q->next = r->next; p->next = r; r->next = q;`

B. `p->next = r; q->next = r->next; r->next = q;`

C. `q->next = r->next; r->next = q; p->next = r;`

D. `r->next = q; q->next = r->next; p->next = r;`

12. 同时查找 2n 个数中的最大值和最小值，最少比较次数为（　）。

A. 3(n-2)/2　　B. 4n-2　　C. 3n-2　　D. 2n-2

13. 设 G 是有 6 个结点的完全图，要得到一棵生成树，需要从 G 中删去（　）条边。

A. 6　　B. 9　　C. 10　　D. 15

14. 以下时间复杂度不是 $O(n^2)$的排序方法是（　）。

A. 插入排序　　B. 归并排序　　C. 冒泡排序　　D. 选择排序

15. 以下程序段实现了找第二小元素的算法。输入是 n 个不等的数构成的数组 S，输出 S 中第二小的数 SecondMin。在最坏情况下，该算法需要做（　）次比较。

```
if (S[1] < S[2]) {
    FirstMin = S[1];
    SecondMin = S[2];
} else {
    FirstMin = S[2];
    SecondMin = S[1];
}
for (i = 3; i <= n; i++)
    if (S[i] < SecondMin)
        if (S[i] < FirstMin) {
            SecondMin = FirstMin;
            FirstMin = S[i];
        } else {
            SecondMin = S[i];
        }
```

A. 2n　　B. n-1　　C. 2n-3　　D. 2n-2

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. 若逻辑变量 A、C 为真，B、D 为假，以下逻辑运算表达式为真的有（　　）。

A. (B ∨ C ∨ D) ∨ D ∧ A　　B. ((¬A ∧B) ∨ C) ∧ ¬B

C. (A ∧ B) ∨ (C ∧ D ∨ ¬A)　　D. A ∧ (D ∨ ¬C) ∧ B

2. 下列（　　）软件属于操作系统软件。

A. Microsoft Word

B. Windows XP

C. Android

D. Mac OS X

E. Oracle

3. 在 NOI 比赛中，对于程序设计题，选手提交的答案不得包含下列哪些内容（　　）。

A. 试图访问网络

B. 打开或创建题目规定的输入/输出文件之外的其他文件

C. 运行其他程序

D. 改变文件系统的访问权限

E. 读写文件系统的管理信息

4. 以下哪些结构可以用来存储图（　）。

A. 邻接矩阵　　B. 栈　　C. 邻接表　　D. 二叉树

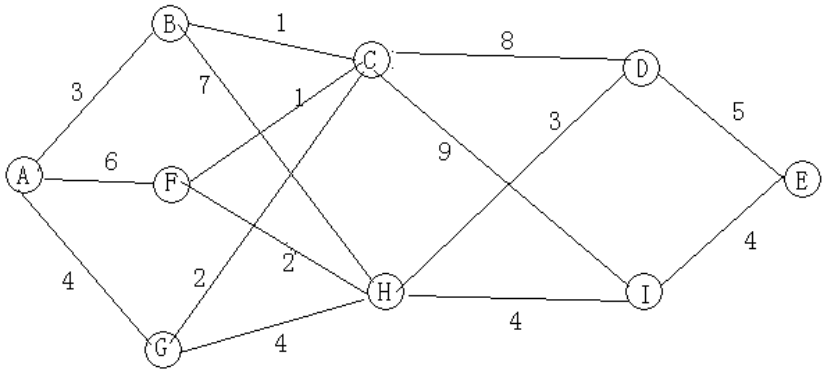
5. 下列各无符号十进制整数中，能用八位二进制表示的数有（　）。

A. 296　　B. 133　　C. 256　　D. 199

三、问题求解（共 2 题，每题 5 分，共计 10 分；每题全部答对得 5 分，没有部分分）

1. 由数字 1,1,2,4,8,8 所组成的不同的四位数的个数是_____。

2. 如图所示，图中每条边上的数字表示该边的长度，则从 A 到 E 的最短距离是_____。



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 main() {
    int a, b, i, tot, c1, c2;
    cin >> a >> b;
    tot = 0;
    for (i = a; i <= b; i++)
    {
        c1 = i / 10;
        c2 = i % 10;
```

```
        if ((c1 + c2) % 3 == 0)
            tot++;
    }
    cout << tot << endl;
    return 0;
}
```

输入：7 31

输出：_____

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

int fun(int n, int minNum, int maxNum) {
    int tot, i;
    if (n == 0)
        return 1;
    tot = 0;
    for (i = minNum; i <= maxNum; i++)
        tot += fun(n - 1, i + 1, maxNum);
    return tot;
}

int main() {
    int n, m;
    cin >> n >> m;
    cout << fun(m, 1, n) << endl;
    return 0;
}
```

输入：6 3

输出：_____

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

```
const int SIZE = 100;

int main() {
    string dict[SIZE];
    int rank[SIZE];
    int ind[SIZE];
    int i, j, n, tmp;
    cin >> n;
    for (i = 1; i <= n; i++) {
        rank[i] = i;
        ind[i] = i;
        cin >> dict[i];
    }
    for (i = 1; i < n; i++)
        for (j = 1; j <= n - i; j++)
            if (dict[ind[j]] > dict[ind[j + 1]]){
                tmp = ind[j];
                ind[j] = ind[j + 1];
                ind[j + 1] = tmp;
            }
    for (i = 1; i <= n; i++)
        rank[ind[i]] = i;
    for (i = 1; i <= n; i++)
        cout << rank[i] << " ";
    cout << endl;
    return 0;
}
```

输入：

```
7
aaa
aba
bbb
aaa
aaa
ccc
aa
```

输出：_____

4.

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

const int SIZE = 100;

int alive[SIZE];
int n;

int next(int num) {
    do {
        num++;
        if (num > n)
            num = 1;
    } while (alive[num] == 0);
    return num;
}

int main() {
    int m, i, j, num;
    cin >> n >> m;
    for (i = 1; i <= n; i++)
        alive[i] = 1;
    num = 1;
    for (i = 1; i <= n; i++) {
        for (j = 1; j < m; j++)
            num = next(num);
        cout << num << " ";
        alive[num] = 0;
        if (i < n)
            num = next(num);
    }
    cout << endl;
    return 0;
}
```

输入：11 3

输出：_____

五、完善程序（每题 14 分，共计 28 分）

1. （**双栈模拟数组**）只使用两个栈结构 stack1 和 stack2，模拟对数组的随机读取。作为栈结构，stack1 和 stack2 只能访问栈顶（最后一个有效元素）。栈顶指针 top1 和 top2 均指向栈顶元素的下一个位置。

输入第一行包含两个整数，分别是数组长度 n 和访问次数 m，中间用单个空格隔开。第二行包含 n 个整数，依次给出数组各项（数组下标从 0 到 n-1）。第三行包含 m 个整数，需要访问的数组下标。对于每次访问，输出对应的数组元素。（前两空每空 2.5 分，其余每空 3 分，共 14 分）

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

const int SIZE = 100;

int stack1[SIZE], stack2[SIZE];
int top1, top2;
int n, m, i, j;

void clearStack() {
    int i;
    for (i = top1; i < SIZE; i++)
        stack1[i] = 0;
    for (i = top2; i < SIZE; i++)
        stack2[i] = 0;
}

int main() {
    cin >> n >> m;
    for (i = 0; i < n; i++)
        cin >> stack1[i];
    top1 = ___(1)___;
    top2 = ___(2)___;
    for (j = 0; j < m; j++) {
```

```
        cin >> i;
        while (i < top1 - 1) {
            top1--;
            ___(3)___;
            top2++;
        }
        while (i > top1 - 1) {
            top2--;
            ___(4)___;
            top1++;
        }
        clearStack();
        cout << stack1[___(5)___] << endl;
    }
    return 0;
}
```

2. **（最大子矩阵和）**给出 m 行 n 列的整数矩阵，求最大的子矩阵和（子矩阵不能为空）。

输入第一行包含两个整数 m 和 n，即矩阵的行数和列数。之后 m 行，每行 n 个整数，描述整个矩阵。程序最终输出最大的子矩阵和。（第一空 2 分，其余 3 分，共 14 分）

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

const int SIZE = 100;

int matrix[SIZE + 1][SIZE + 1];
int rowsum[SIZE + 1][SIZE + 1];    //rowsum[i][j]记录第 i 行前 j 个数的和
int m, n, i, j, first, last, area, ans;

int main() {
    cin >> m >> n;
    for (i = 1; i <= m; i++)
        for (j = 1; j <= n; j++)
            cin >> matrix[i][j];
```

```
ans = matrix[0][0];
for (i = 1; i <= m; i++)
    (2);
    for (i = 1; i <= m; i++)
        for (j = 1; j <= n; j++)
            rowsum[i][j] = (3);
    for (first = 1; first <= n; first++)
        for (last = first; last <= n; last++) {
            (4);
            for (i = 1; i <= m; i++) {
                area += (5);
                if (area > ans)
                    ans = area;
                if (area < 0)
                    area = 0;
            }
        }
    cout << ans << endl;
    return 0;
}
```