

7. 与二进制小数 0.1 相等的十六进制数是（ ）。
- A. 0.8 B. 0.4 C. 0.2 D. 0.1
8. 所谓的“中断”是指（ ）。
- A. 操作系统随意停止一个程序的运行
B. 当出现需要时，CPU 暂时停止当前程序的执行转而执行处理新情况的过程
C. 因停机而停止一个程序的运行
D. 电脑死机
9. 计算机病毒是（ ）。
- A. 通过计算机传播的危害人体健康的一种病毒
B. 人为制造的能够侵入计算机系统并给计算机带来故障的程序或指令集合
C. 一种由于计算机元器件老化而产生的对生态环境有害的物质
D. 利用计算机的海量高速运算能力而研制出来的用于疾病预防的新型病毒
10. 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
11. 下面哪种软件不属于即时通信软件（ ）。
- A. QQ B. MSN C. 微信 D. P2P
12. 6 个顶点的连通图的最小生成树，其边数为（ ）。
- A. 6 B. 5 C. 7 D. 4
13. 链表不具备的特点是（ ）。
- A. 可随机访问任何一个元素
B. 插入、删除操作不需要移动元素
C. 无需事先估计存储空间大小
D. 所需存储空间与存储元素个数成正比
14. 线性表若采用链表存储结构，要求内存中可用存储单元地址（ ）。
- A. 必须连续 B. 部分地址必须连续
C. 一定不连续 D. 连续不连续均可
15. 今有一空栈 S，对下列待进栈的数据元素序列 a,b,c,d,e,f 依次进行进栈，进栈，出栈，进栈，进栈，出栈的操作，则此操作完成后，栈 S 的栈顶元素为（ ）。
- A. f B. c C. a D. b

16. 前序遍历序列与中序遍历序列相同的二叉树为（ ）。
- A. 根结点无左子树的二叉树
 - B. 根结点无右子树的二叉树
 - C. 只有根结点的二叉树或非叶子结点只有左子树的二叉树
 - D. 只有根结点的二叉树或非叶子结点只有右子树的二叉树
17. 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
18. 下列选项中不属于视频文件格式的是（ ）。
- A. TXT
 - B. AVI
 - C. MOV
 - D. RMVB
19. 设某算法的计算时间表示为递推关系式 $T(n) = T(n-1) + n$ (n 为正整数) 及 $T(0) = 1$ ，则该算法的时间复杂度为（ ）。
- A. $O(\log n)$
 - B. $O(n \log n)$
 - C. $O(n)$
 - D. $O(n^2)$
20. 在 NOI 系列赛事中参赛选手必须使用由承办单位统一提供的设备。下列物品中不允许选手自带的是（ ）。
- A. 鼠标
 - B. 笔
 - C. 身份证
 - D. 准考证

二、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. 重新排列 1234 使得每一个数字都不在原来的位置上，一共有_____种排法。
2. 一棵结点数为 2015 的二叉树最多有_____个叶子结点。

三、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, c;
    a = 1;
    b = 2;
    c = 3;
```

```

    if (a > b) {
        if (a > c)
            cout << a << ' ';
        else
            cout << b << ' ';
    }
    cout << c << endl;
    return 0;
}

```

输出: _____

2. #include <iostream>
using namespace std;
struct point {
 int x;
 int y;
};
int main() {
 struct EX {
 int a;
 int b;
 point c;
 } e;
 e.a = 1;
 e.b = 2;
 e.c.x = e.a + e.b;
 e.c.y = e.a * e.b;
 cout << e.c.x << ',' << e.c.y << endl;
 return 0;
}

输出: _____

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

```

int main() {
    string str;
    int i;
    int count;
    count = 0;
    getline(cin, str);
    for (i = 0; i < str.length(); i++) {
        if(str[i] >= 'a' && str[i] <= 'z')
            count++;
    }
    cout << "It has " << count << " lowercases" << endl;
    return 0;
}

```

输入: NOI2016 will be held in Mian Yang.

输出: _____

4. #include <iostream>

```

using namespace std;
void fun(char *a, char *b) {
    a = b;
    (*a)++;
}
int main() {
    char c1, c2, *p1, *p2;
    c1 = 'A';
    c2 = 'a';
    p1 = &c1;
    p2 = &c2;
    fun(p1, p2);
    cout << c1 << c2 << endl;
    return 0;
}

```

输出: _____

四、完善程序（共 2 题，每题 14 分，共计 28 分）

1. （打印月历）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。（第三、四空 2.5 分，其余 3 分）

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
```

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

const int dayNum[]={-1, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
int m, offset, i;

int main() {
    cin >> m;
    cout << "S\tM\tT\tW\tT\tF\tS" << endl; // '\t'为 TAB 制表符
    (1);
    for (i = 1; i < m; i++)
        offset = (2);
    for (i = 0; i < offset; i++)
        cout << '\t';
    for (i = 1; i <= (3); i++) {
        cout << (4);
        if (i == dayNum[m] || (5) == 0)
            cout << endl;
        else
            cout << '\t';
    }
    return 0;
}
```

2. (中位数) 给定 n (n 为奇数且小于 1000) 个整数, 整数的范围在 $0 \sim m$ ($0 < m < 2^{31}$) 之间, 请使用二分法求这 n 个整数的中位数。所谓中位数, 是指将这 n 个数排序之后, 排在正中间的数。(第五空 2 分, 其余 3 分)

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

const int MAXN = 1000;

int n, i, lbound, rbound, mid, m, count;
int x[MAXN];

int main() {
    cin >> n >> m;
    for (i = 0; i < n; i++)
        cin >> x[i];
    lbound = 0;
    rbound = m;
    while ( (1) ) {
        mid = (lbound + rbound) / 2;
        (2);
        for (i = 0; i < n; i++)
            if ( (3) )
                (4);
        if (count > n / 2)
            lbound = mid + 1;
        else
            (5);
    }
    cout << rbound << endl;
    return 0;
}
```