CSP-J 模拟（六）

2025年8月16日

一、单项选择题

- 共 15 题，每题 2 分，共计 30 分
- 每题有且仅有一个正确选项

1. 下列无符号数中，最小的数是（　　），下标表示该数的进制。
   A. $(1011001)_2$
   B. $(75)_{10}$
   C. $(37)_8$
   D. $(2A)_{16}$

2. 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. $(A \wedge B) \vee (C \wedge D \vee \neg A)$
   B. $(B \vee C \vee D) \vee D \wedge A$
   C. $A \wedge (D \vee \neg C) \wedge B$
   D. $(D \vee \neg C) \wedge \neg B \wedge \neg C$

3. 已知 a, b, c, d, e, f, g 七个人中，a 会讲英语，b 会讲英语和汉语，c 会讲英语、意大利语和俄语，d 会讲汉语和日语，e 会讲意大利语和德语，f 会讲俄语、日语和法语，g 会讲德语和法语。将他们的座位安排在圆桌旁，有多少种本质不同的排列方法可以使每个人都能与他身边的人交流（　　）。

   A. 1

   B. 2

   C. 3

   D. 4

4. 如果对一个已经升序的数组进行排序，下列算法中可能花费时间反而比乱序数组多的是（　　）
   A. 堆排序
   B. 插入排序
   C. 冒泡排序
   D. 快速排序

5. 按中序遍历二叉树的结果为 ABC，有（　　）种不同形态的二叉树可以得到这一遍历结果。
   A. 3
   B. 4
   C. 5
   D. 6

6. 设一个栈与一个队列的初始状态均为空。元素 1、2、3、4、5、6 依次进入栈，且每个元素出栈后即进入队列，若出队的顺序为 2、4、3、6、5、1，则栈的容量至少应该为（　　）。
   A. 2
   B. 3
   C. 4
   D. 5

7. 整型数组 a 中有 n 个元素，能计算 a 中有多少个数字大于 lower 且小于 upper 的函数，应该将下划线依次替换为：

```
int solve(int a[], int n, int lower, int upper)
{
   std::sort(a, a + n);
   auto begin = std::_______(a, a + n, lower);
   auto end = std::_______(a, a + n, upper);
   return end - begin;
}
```

A. lower_bound 、 lower_bound

B. lower_bound 、 upper_bound

C. upper_bound 、 lower_bound

D. upper_bound 、 upper_bound

8. 二叉树 T 的广度优先遍历序列为A、B、C、D、E、F、G、H、I，已知A是C的父节点，D是G的父节点，F是I的父节点，树中所有节点的最大深度为3，根节点的深度为0，可知E的父节点可能是（ ）。

A. A, B

B. B, C

C. A, B, F

D. B, C, E

9. 已知字符集a、b、c、d、e、f、g、h。若各字符的哈夫曼编码依次是 0100，10，0000，0101，001，011，11，0001，则编码序列 0100011001001011110101 的译码结果是（ ）。

A. acgabfh

B. adbagbb

C. afbeagd

D. afeefgd

10. 字符串 ababacbab 和字符串 abcba 的最长公共子串（注意子串与子序列的区别）是（ ）

A. abcba

B. cba

C. abc

D. bcba

11. 设有一个含有 13 个元素的哈希表 (0 12)，哈希函数是：H(key) = key % 13。用线性探查法解决冲突，则对于序列 {2、8、31、20、19、18、53、27}，18应放在下标为（ ）的位置。

A. 0

B. 4

C. 5

D. 9

12. 无向图 G =(V,E)，期中 V={a,b,c,d,e,f}，E ={(a,b),(a,e),(a,c),(b,e),(c,f),(f,d),(e,d)}，对该图进行深度优先遍历，得到的顶点序列正确的是（ ）

A. a, b, e, c, d, f

B. a, c, f, e, b, d

C. a, e, b, c, f, d

D. a, b, e, d, f, c

13. 平面上有三条平行直线，每条直线上分别有5, 6, 7个点，且不同直线上的三个点都不在同一条直线上。用这些点为顶点，能组成多少个不同的四边形（　　）。

A. 210

B. 751

C. 1495

D. 2250

14. 已知 p 为 int 类型，q 为 int * 类型，不符合语法的是（　）

A. `*q = p;`

B. `p = *q;`

C. `*(p + q) = p;`

D. `*(p + q) = q;`

15. 计算机能直接执行的指令包括两部分，它们是（　）。

A. 源操作数与目标操作数

B. 操作码与操作数

C. ASCII码与汉子代码

D. 数字与字符

二、阅读程序

- 判断题 1 分，选择题 3 分，共计 40 分
- 判断题正确填 T，错误填 F；

第1题

```
int solve1(int n)
{
    int s = 0;
    for (int i = 1; i <= n; ++i) {
        int f = 1;
        for (int j = i; j >= 1; --j) {
            f = f * j;
        }
        s = s + f;
    }
    return s;
}

int solve2(int n)
{
    int s = 0;
    for (int i = n; i >= 1; --i)
    {
        s = s + 1;
        s = s * i;
    }
    return s;
}
```

保证 solve1 与 solve2 的参数 n 是正整数。

判断题

16. solve1(4) 的返回值为 33 （ ）。

17. solve2(3) 的返回值为 10 （ ）

选择题

18. 关于 solve1(n) 与 solve2(n) 的大小关系，下列说法正确的是（ ）。

A. 必然有 solve1(n) < solve2(n)
B. 必然有 solve1(n) == solve2(n)
C. 必然有 solve1(n) > solve2(n)
D. 两者大小关系不确定

19. 两个函数都有唯一的从大循环到小的 for 语句，如果将这两条语句的循环次序颠倒，则返回值（ ）。

A. solve1 不变，solve2 变大
B. solve1 不变，solve2 变小
C. solve1 变小，solve2 变小
D. solve1 变大，solve2 变小

20. solve1(n) 与 solve2(n) 的时间复杂度为（ ）。

A. $\Theta(n)$、$\Theta(n)$
B. $\Theta(n)$、$\Theta(n^2)$
C. $\Theta(n^2)$、$\Theta(n)$
D. $\Theta(n^2)$、$\Theta(n^2)$

第2题

```
bool move(int b[], int n)
{
    for (int i = 0; i < n; ++i) {
        if (b[i] == 1) {
            b[i] = 0;
        }
        else {
            b[i] = 1;
            return true;
        }
    }
    return false;
}

void print(int n)
{
    int b[n];
    for (int i = 0; i < n; ++i) b[i] = 0;
    do {
        for (int i = 0; i < n; ++i) std::cout << b[i];
        std::cout << "\n";
```

```
    }
    while (move(b, n));
}

int count(int n)
{
    int b[n];
    for (int i = 0; i < n; ++i) b[i] = 0;
    int c = 0;
    do {
        c++;
    }
    while (move(b, n));
    return c;
}
```

判断题

21. `count` 可能会陷入无尽的循环（ ）。
22. `print` 输出的 `0` 与 `1` 必然一样多（
23. `print(6)` 输出的第 5 行第 4 列的字符为 `0`（ ）
24. `print(16)` 输出的第 523 行第 14 列的字符为 `1`（ ）。

选择题

25. `count(2)` 的返回值为（ ）。

    A. 1
    B. 2
    C. 3
    D. 4

26. `print(n)` 与 `count(n)` 的时间复杂度为（ ）。

    A. $\Theta(2^n)$, $\Theta(2^n)$
    B. $\Theta(n \cdot 2^n)$, $\Theta(2^n)$
    C. $\Theta(n \cdot 2^n)$, $\Theta(n \cdot 2^n)$
    D. $\Theta(4^n)$, $\Theta(2^n)$

27. 以下说法错误的是（ ）

    A. `print(n)` 前一半输出全是偶数，后一半输出全是奇数
    B. 若将 `print(n)` 输出的每行内容看成一个数字，则他们是依次递增的
    C. `print(n)` 输出的每一行内容都是不同的
    D. `print(n)` 输出的每一列内容都是不同的

第三题

```
struct flat_map {
  struct {
    int key;
    int value;
```

```
} bucket[65536];
int size = 0;

struct result {
  int index;
  bool hit;
};

result find(int begin, int end, int key) {
  if (begin == end)
    return {begin, false};
  else {
    int mid = begin + (end - begin) / 2;
    if (key < bucket[mid].key)
      return find(begin, mid, key);
    else if (bucket[mid].key < key)
      return find(mid+1, end, key);
    else
      return {mid, true};
  }
}

int get(int key) {
  result p = find(0, size, key);
  if (p.hit)
    return bucket[p.index].value;
  else
    return 0;
}

void put(int key, int value) {
  result p = find(0, size, key);
  for (int i = size; i > p.index; --i) {
    bucket[i] = bucket[i - 1];
  }
  size++;
  bucket[p.index].key = key;
  bucket[p.index].value = value;
}
};
```

判断题

28. flat_map 实现了一种关系型容器（ ）。

29. flat_map 对按照键的大小顺序，将键与值配对，存储到了一块连续的内存序列里（ ）。

30. 若数组 bucket 的下标在 [b, e) 范围内存在键 k，find(b, e, k) 函数返回的 hit 为 false（ ）。

31. 当调用 get(k) 后发现 k 不存在，flat_map 会为 k 分配一块内存并将它对应的值置为 0（ ）。

选择题

32. 记 s 表示容器的大小 `size`，调用 `get(key)` 的最坏时间复杂度为（　　）
    A. $\Theta(s)$
    B. $\Theta(\log s)$
    C. $\Theta(s \log s)$
    D. $\Theta(s^2)$

33. 记 s 表示容器的大小 `size`，调用 `put(key, value)` 的最坏时间复杂度为（　　）。
    A. $\Theta(s)$
    B. $\Theta(\log s)$
    C. $\Theta(s \log s)$
    D. $\Theta(s^2)$

34. 以下说法错误的是（　）
    A. `flat_map` 的优点是插入数据时移动数据少。
    B. `flat_map` 的优点是数据紧凑排列，空间使用效率高。
    C. `flat_map` 的优点是查找数据的效率高。
    D. `flat_map` 的优点是代码紧凑，实现简短。

35. 若调用 `put(k, v)` 时已经存在相同的键 `k` 时，以下哪一种处理策略最符合上述代码的逻辑（　）。
    A. 熔断（Breaker）：掷出异常，向系统报告错误
    B. 回滚（Rollback）：不做任何修改，撤销 `put` 操作
    C. 覆盖（Rewrite）：将键所对应的老值覆盖成新值
    D. 忽略（Ignore）：对有可能出错的操作置之不理

三、完善程序

- 单选题，每小题 3 分，共计 30 分

第1题

给定含有 n 个顶点的有向完全图（顶点编号为 0 到 $n-1$）。顶点 x 到 y 的边的权重为 `g[x][y]`。

请找出一条不重复经过任何点的路径，从顶点 0 出发到顶点 $n-1$ 结束，路径上所有边的权重的异或值尽可能大。

```
int n, m;
long long g[MAXN][MAXN];
bool visited[MAXN] = {false};

int dfs(int node, int path) {
    if (____(1)____)
    {
        return ____(2)____;
    }

    int best = 0;
    visited[node] = true;
    for (int next = 0 ; next < ____(3)____; ++next)
    {
        if (____(4)____)
        {
```

```
            best = std::max(best, ____(5)____);
        }
    }
    visited[___(6)___] = ___(7)___;
}

int solve()
{
    return ____(8)_____;
}
```

36. (1) (2) 处应填（ ）。
    A. node == n-1, path
    B. node == n-1, 0
    C. node < n, path
    D. node > n, 1

37. (3) (4)处应填（ ）。
    A. n , visited[next]
    B. n , !visited[next]
    C. m , visited[next]
    D. m , !visited[next]

38. (5) 处应填 ）。
    A. dfs(next, path ^ g[next][node]);
    B. dfs(next, path ^ g[node][next]);
    C. dfs(node + 1, path ^ g[next][node]);
    D. dfs(node + 1, path ^ g[node][next]);

39. (6) (7)处应填（ ）。
    A. node , false
    B. node , true
    C. next , false
    D. next , true

40. (8)处应填（
    A. dfs(0, 0)
    B. dfs(1, 0)
    C. dfs(0, 1)
    D. dfs(1, 1)

第2题

n 个岛屿由 n 座桥连成环。岛的编号为 0 到 $n-1$，第 i 座桥连第 i 号岛与第 $(i+1) \bmod n$ 号岛。

某旅行团从第 x_1 号岛出发，依次访问的岛编号为 $x_2, \ldots, x_m$。

请选择拆掉一座桥，使得旅行团的过桥次数达到最小，令 solve 函数返回这个最小值。

```
int solve(int n, int m, int x[])
{
    int diff[n];
```

```
    for (int i = 0; i < n; ++i) diff[i] = 0;
    int base = 0;
    for (int i = 1; i < m; ++i)
    {
        int prev = x[i - 1];
        int next = x[i];
        int begin, end;
        if (prev < next) {
            begin = prev;
            end = next;
        }
        else {
            begin = next;
            end = prev;
        }
        base += ____(1)____;
        int inc = ____(2)____ ;
        diff[____(3)____] += inc;
        diff[____(4)____] -= inc;
        prev = next;
    }
    int best = n * m;
    int sum = 0;
    for (int i = 0; i < n; ++i)
    {
        ____(5)____;
        if (best > sum) best = sum;
    }
    return ____(6)_____;
}
```

41. (1)处应填（ ）。

A. end - begin

B. end - begin - 1

C. end - begin + 1

D. end - begin - 2

42. (2)处应填（

A. base

B. end*2 - begin*2

C. begin*2 - end*2

D. n + begin*2 - end*2

43. (3)(4)处应填（ ）。

A. begin , end

B. begin , end + 1

C. end , begin

D. end + 1 , begin

44 (5)处应填（

A. `sum += diff[i]`

B. `sum += diff[i + 1]`

C. `sum += diff[i + 1] - diff[i]`

D. `sum += diff[i] - diff[i-1]`

45. (6)处应填（

A. `sum`

B. `best`

C. `best - base`

D. `best + base`