2024 LUOGU 非专业级别收容能力认证第一轮

（SCP-S1）提高级 C++语言试题

认证时间：2024 年 8 月 18 日 14:30~16:30

考生注意事项：

- 试题纸共有 12 页，满分 100 分。请在洛谷作答，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。
- 试题由洛谷网校学术组命制，欢迎报名洛谷网校第一轮课程。课程内容包含专题讲解、真题讲评与本试题讲评。https://class.luogu.com.cn/course/yugu24acs

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．下列编译命令，是 CSP-S 第二轮评测时用于编译题目英文名为 luogu 的传统型试题的选手代码的是（　）。

A．`gcc -std=c++14 -O2 -o * luogu.cpp`

B．`g++ -std=c++14 -O2 -o * luogu.cpp`

C．`g++ -O2 -o * luogu.cpp`

D．`g++ -std=c++14 -o * luogu.cpp`

2．在 2023 年的 CSP-S 第二轮认证中，有四位同学对于某一题得 0 分的情况提出了申诉。根据 CCF NOI 竞赛委员会的相关规定，下列申诉中可能被接受的是（　）。

A．小明在提交的程序中并未删除用于输出调试信息的代码。

B．小红在提交的程序中使用了随机化算法。

C．小黄提交的程序在考场的 Windows 电脑上正常编译，而在最终评测时编译失败。

D．小陈在与评测环境相同的电脑上运行程序，使用了 850 毫秒运行得出正确答案，而该题的时间限制为 1 秒。程序运行并未超出内存限制。

3．大型语言模型（LLM）指使用大量文本数据训练的深度学习模型，它们可以生成自然语言文本或理解语言文本的含义。下列人工智能技术的应用中，不属于 LLM 的是（　）。

A．ChatGPT

B．Stable Diffusion

C．Gemini

D．文心一言

4．$(47)_{10}$ 与下列（　）选项表示的数值一致？

A．$(56)_8$

B．$(1100110)_2$

C．$(142)_5$

D．$(2D)_{16}$

5．定义 a 在模 p 意义下的乘法逆元为同余方程 $ax \equiv 1 (\mathrm{mod}\, p)$ 的解 x。请计算 13 在模 29 意义下的乘法逆元是（　）。

A．11

B．10

C．8

D．9

6．仿照二项式系数，定义三项式系数 T_n^k 为 $(1+x+x^2)^n$ 的展开式中 x^k 项的系数。即：$(1+x+x^2)^n = T_n^0 + T_n^1 x^1 + T_n^2 x^2 + \cdots + T_n^{2n} x^{2n}$。现要计算 $T_8^0 + T_8^1 + T_8^2 + \cdots + T_8^{16}$ 的值，其结果为（　）？

A．4096

B．6561

C．8192

D．15625

7．当在二叉排序树中插入一个新结点时，若树中不存在与待插入结点关键字相同的结点，且待插入结点的关键字小于根结点的关键字，则新结点将成为根结点的（　）？

A．左子树的某一叶子结点

B．右子树的某一叶子结点

C．左子树的某一非叶子结点

D．右子树的某一非叶子结点

8．一个盒子里有 4 个红球、5 个蓝球和 6 个绿球。现在从盒子里随机抽取 3 个球，则这 3 个球中恰好有 1 个红球、1 个蓝球和 1 个绿球的概率是（　）。

A．136/455

B．24/91

C．5/66

D．1/11

9．对前缀表达式 `- + 10 * 2 3 + 5 / 6 2` 进行求值，得到的结果是（　）。

A．8

B．9

C．10

D．11

10．已知一个循环队列的存储空间为数组 `data q[21]`且不浪费空间。且在队列的定义中，头指针指向队列的开头，尾指针指向队列末尾的下一个元素。现在若头指针和尾指针分别指向下标 8 和 3，则队列当前的长度为（　）？

A．5

B．6

C．17

D．16

11．在使用下列算法对整数数列进行排序时，哪一个算法的运行时间复杂度与整数大小无关（ ）。

A．基数排序

B．计数排序

C．希尔排序

D．以上排序算法的运行时间复杂度均与整数大小有关

12． 假设变量 a,b,c 均为绝对值不超过 10^9 的 32 位有符号整型变量，下列关于这三个变量，说法正确的是（ ）。

A．若执行代码 `auto d = a * b;` 则 d 的变量类型是 64 位有符号整型变量。

B．逻辑表达式 `max(a, b) * 1LL * c == max(1LL * a * c, 1LL * b * c)` 的运算结果一定为真。

C．执行代码 `cout << sizeof(a);` 后，输出的结果为 4。

D．`for (unsigned i = a; i < b; i++)` 可能是一个死循环。

13．对于以下代码，假设执行单次 `rand()`函数的时间复杂度为 `O(1)`，那么 `calc` 函数的运行时间复杂度为？（ ）

```
unsigned int ans = 0;
void val(int n) {
    for (int i = 1; i <= n; ++ i)
        for(int j = 0; j < n; j += i)
            ans += rand();
}
void calc(int n) {
    if (n <= 1) val(n);
    else calc(n / 2), calc(n / 2), val(n);
}
```

A．$\Theta(n)$

B．$\Theta(n \log n)$

C．$\Theta(n \log^2 n)$

D．$\Theta(n \log^3 n)$

14．给定一个无向图 G，其中包含以下顶点和边：

顶点集合：`V = {A, B, C, D, E}`

边集合：`E = {(A, B), (A, C), (B, C), (B, D), (C, D), (C, E)}`

请问以下哪一个选项正确描述了图 G 的双连通性？（　）。

A．图 G 是点双连通的，因为删除任意一个顶点后，图仍然连通。

B．图 G 不是点双连通的，因为存在割点。

C．图 G 是边双连通的，因为存在至少两个顶点的简单环。

D．图 G 不是边双连通的，因为删除任意一条边后，图不再连通。

15．使用哈希表存储元素 19,53,49,114，为了让哈希表不发生哈希冲突，可以使用（　）选项的哈希函数？（定义：$\lfloor x \rfloor$ 为不超过 x 的最大整数，如 $\lfloor 3.14 \rfloor = 3$）

A．$f(x) = x \bmod 17$

B．$f(x) = \lfloor \sqrt{x} \rfloor$

C．$f(x) = \lfloor 100 \lg x \rfloor$

D．$f(x) = x^2 \bmod 13$

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 T，错误填 F；除特殊说明外，判断题 1.5 分，选择题 3 分，共计 40 分）

（1）

```
#include <bits/stdc++.h>
using namespace std;
set <int> s1 = {1, 2, 3}, s2 = {2, 3, 1, 1},
          s3 = {1, 2, 100}, s4 = {3, 4, 5};
map <set <int>, int> mp1, mp2;
set <int> intersection(set <int> a, set <int> b) {
   if (a.size() > b.size()) swap(a, b);
   set <int> res = a;
   for (int i : a)
      if (b.count(i) == 0)
         res.erase(i);
   return res;
}
int main() {
   cout << (s1 == s2) << endl;
   mp1[s3] = 9; mp1[s1] = 1;
   mp2[s2] = 1; mp2[s4] = 5;
   cout << (mp1 < mp2) << ' ';
   cout << mp2[s3] << ' ';
   cout << (mp1 <= mp2) << endl;
```

```
21    mp1.clear();
22    int n, l, t;
23    cin >> n;
24    for (int i = 1; i <= n; i++) {
25       set <int> tmp;
26       for (cin >> l; l; l--) {
27          cin >> t;
28          tmp.emplace(t);
29       }
30       mp1[tmp] = i;
31    }
32    set <int> res = mp1.begin()->first;
33    for (auto i : mp1)
34       res = intersection(i.first, res);
35    cout << res.size();
36    return 0;
37 }
```

假设输入的 n,l,t 满足：n 是不超过 100 的正整数，l 和 t 是不超过 1000 的正整数，完成下面的判断题和单选题。

• 判断题

16. 代码运行后，输出的第一行是 0。（ ）

17. 设集合 a,b 的元素个数分别为 p,q，则执行函数 intersection(a,b)的时间复杂度为 $\Theta(\min(p,q)\log\max(p,q))$。（ ）

18. 将第 32 行的 begin()换成 rbegin()，运行结果可能改变。（ ）

19. 将第 33 行的 auto 换成 pair <set <int>, int>，运行结果不变。（ ）

• 单选题

20. 输出的第二行是（ ）。

A. 0 0 0　　B. 1 0 1　　C. 0 1 1　　D. 1 0 0

21. 记所有输入的 l 之和为 L，则该程序最坏情况下的时间复杂度是（ ）？

A. $\Theta(L\log L)$　　B. $\Theta(L\log^2 L)$

C. $\Theta(nL\log L)$　　D. $\Theta(nL\log n\log L)$

（2）

```
#include <bits/stdc++.h>
using namespace std;
constexpr int mod = 998244353;
int n, k, a[12], b[12], ans[12], fa[12];
int findfa(int u) { return u == fa[u] ? u : fa[u] = findfa(fa[u]); }
int functionUnknown(int a[], int n) {
    if (n <= 1) return 0;
    int i = n - 1, j, k;
    while (true) {
        j = i; --i;
        if (a[i] < a[j]) {
            for (k = n; a[i] >= a[--k];);
            swap(a[i], a[k]);
            reverse(a + j, a + n);
            return 1;
        }
        if (!i) {
            reverse(a, a + n);
            return 0;
        }
    }
    return -1;
}
int F(int x) {
    int ans = 0;
    for (int i = k - 1; ~i; --i)
        ans = (1ll * ans * x % mod + a[i]) % mod;
    return ans;
}
int main() {
    scanf("%d %d", &n, &k);
    for (int i = 0; i < k; ++i) scanf("%d", a + i);
    for (int m = 1; m <= n; ++m) {
        for (int i = 0; i < m; ++i) b[i] = i;
        do {
            for (int i = 0; i < m; ++i) fa[i] = i;
            int res = m;
```

```
38            for (int i = 0, u, v; i < m; ++i) {
39                u = findfa(i); v = findfa(b[i]);
40                if (u == v) continue;
41                --res; fa[u] = v;
42            }
43            int flag = 0;
44            for (int i = 0; i < m; ++i)
45                if (b[i] == i) flag = 1;
46            if (flag) continue;
47            ans[m] = (ans[m] + F(res)) % mod;
48        } while(functionUnknown(b, m));
49        printf("%d%c",ans[m]," \n"[m == n]);
50    }
51    return 0;
52 }
```

假设输入的 n,k 是不超过 10 的正整数，输入的 a[i]是不超过 10^8 的正整数，完成下面的判断题和单选题：

• 判断题

22. functionUnknown 函数的返回值不可能为 -1。（ ）

23. 当输入的 k=1 时，输出的数值均能表示为 $t \times a[0]$ 的形式，其中 t 是一个整数。（ ）

24. 如果把第 34 行**整行**改为 b[m - 1] = m - 1，则程序输出将会被改变。（ ）

• 单选题

25. 在主函数中，findfa 函数被调用的次数与以下哪个选项同阶（ ）。

A. $\Theta(n!)$　　B. $\Theta(n!\,n)$

C. $\Theta(n!\,n^2)$　　D. $\Theta(n!\,n^3)$

26. 对于以下的输入数据，输出结果为（ ）。

```
5 5
1 2 3 4 5
```

A. 0 15 30 261 1500　　B. 0 12 24 297 1788

C. 0 15 30 477 2940　　D. 0 15 30 864 5520

27. functionUnknown 的功能接近于 C++ STL 库中的以下（ ）函数。

A. next_permutation　　B. prev_permutation

C. is_sorted　　D. is_permutation

（3）

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 unsigned f(unsigned x) {
4    x ^= x << 3;
5    x ^= x >> 5;
6    return x;
7 }
8 unsigned g(unsigned x) {
9    return (x >> 5) ^ (x >> 2) ^ x ^ (x << 3);
10 }
11 int main() {
12    unsigned y, l, r;
13    cin >> y >> l >> r;
14    for (unsigned long long i = l; i <= r; i++)
15       if (f(i) == y)
16          cout << i % 11 << endl;
17    return 0;
18 }
```

除第 28 题外，假设输入的 y,l,r 均为不超过 $2^{32}-1$ 的非负整数，完成下面的判断题和单选题：

• 判断题

28. 若输入 2024 1 -1，那么程序将不会输出任何数字。（ ）

29. 若把第 14 行 i 的类型改为 unsigned int，则程序可能无法结束运行。（ ）

30. 对于所有符合要求的输入，输出的数字不可能超过一个。（ ）

• 选择题

31. 若从所有 unsigned int 当中随机均匀选取一个 x，则 f(x)=g(x)成立的概率和（ ）的差最小。

A. 0　　B. $\frac{1}{15}$　　C. $\frac{1}{2}$　　D. 1

32. （4 分）定义 $f^n(x)$ 表示对 x 用函数 f 作用 n 次的结果，例如 $f^2(x)=f\big(f(x)\big)$。假设 unsigned 类型具有 ω 个二进制位，且已知存在某种计算 $f^n(x)$ 的算法的时间复杂度为 $\Theta\big(T(n)P(\omega)\big)$，其中 $P(\omega)$ 是一个关于 ω 的幂函数。则 $T(n)$ 最小是（ ）？

A. $\Theta(1)$　　B. $\Theta(\log n)$　　C. $\Theta(\log^2 \mathrm{n})$　　D. $\Theta(n)$

33. 当输入为 50 1 4294967295 时，输出的第一个数是（ ）。

A. 1　　B. 8　　C. 9　　D. 10

三、完善程序（单选题，每小题 3 分，共计 30 分）

（1）（**最小垄断集**）问题：你有一张 n 个结点（结点编号从 1 到 n），m 条边的无向无权图 G=(V,E)，其中 V 为点集，E 为边集。称 V 的一个子集 S **垄断**了图 G，当且仅当 $\forall(u,v)\in E$，u,v 中必然**恰有一个结点**在 S 内。求垄断了 G 的子集 S 至少有多少个元素，或报告不存在这样的子集。试补全程序。

```
#include <bits/stdc++.h>
using namespace std;
int n, m, is[100005];
vector <int> g[100005];
queue <int> q;
int main() {
   cin >> n >> m;
   for (int i = 1; i <= m; i++) {
      int u, v;
      cin >> u >> v;
      g[u].emplace_back(v);
      g[v].emplace_back(u);
   }
   int tot = 0;
   for (int k = 1; k <= n; k++) {
      if (①) continue;
      q.emplace(k);
      is[k] = 1;
      int cnt[3] = ②;
      while (q.size()) {
         for (int i = 0; i < g[q.front()].size(); i++) {
            int to = g[q.front()][i];
            if (is[to] == 0) {
               is[to] = ③;
               cnt[is[to]]++;
               q.emplace(to);
            } else if (④) {
               puts("No such subset.");// 报告不存在这样的子集
               return 0;
            }
```

```
31          }
32          q.pop();
33       }
34       tot += ⑤;
35    }
36    cout << tot;
37    return 0;
38 }
```

34. ①处应填（　）

A. is[k]　　B. is[k] == 0　　C. q.empty()　　D. tot > k

35. ②处应填（　）

A. {0, 0, 0}　　B. {0, 0, 1}

C. {1, 0, 0}　　D. {0, 1, 0}

36. ③处应填（　）

A. is[q.front()]　　B. 3 – is[q.front()]

C. !is[q.front()]　　D. q.front()

37. ④处应填（　）。

A. is[to] == is[q.front()]　　B. is[to] != is[q.front()]

C. is[to] && is[q.front()]　　D. abs(is[to] – is[q.front()]) == 1

38. ⑤处应填（　）。

A. cnt[1]　　B. min(cnt[0], cnt[1])

C. min(cnt[1], cnt[2])　　D. cnt[1] + cnt[2]

（2）（最长公共前缀）定义 LCP(s,t)为两个字符串的最长公共前缀。例如：s=abcde，t=abcabc，则 LCP(s,t)=abc。定义|s|为字符串 s 的长度，在上一例中|s|=5。

现在给定了 n 个字符串 $s_1, s_2, \ldots, s_n$。现在需要统计所有同时满足如下两个条件的 $\left|\mathrm{LCP}(s_i, s_j)\right|$ 之和对 998244353 取模的值：条件 1：$i < j$。条件 2：$s_i < s_j$。

若 $1 \le n \le 2 \times 10^5$，$1 \le |s_i| \le 20$，字符串仅由小写字母构成，试补全程序。

```
#include <bits/stdc++.h>
using namespace std;
const int maxn = 2e5 + 9, mod = 998244353;
char s[maxn][22];
int n;
int tmp[27][maxn], a[maxn];
int dfs(int l, int r, int p) {
    if (l >= r || p >= 20) return 0;
    int tail[27], ans = 0;
    memset(tail, 0, sizeof(tail));
    for (int i = l; i <= r; ++i) {
        int ch = ①;
        ②;
        for (int j = 0; j < ch; ++j)
            ③;
    }
    int tot = l,pos = l;
    for (int i = 0; i < 27; ++i)
        for (int j = 0; j < tail[i]; ++j)
            a[tot++] = tmp[i][j];
    for (int i = 0; i < 27; ++i) {
        ④;
        pos += tail[i];
    }
    return ans;
}
int main() {
    scanf("%d", &n);
    for (int i = 1; i <= n; ++i) scanf("%s", s[i]);
    for (int i = 1; i <= n; ++i) a[i] = i;
    printf("%d\n", ⑤);
    return 0;
```

```
33 }
```

39. ①处可以填（　）

A. `s[i][p] ? s[i][p] - 'a': 0`

B. `s[i][p] ? s[i][p] - 'a' + 1 : 0`

C. `s[a[i]][p] ? s[a[i]][p] - 'a': 0`

D. `s[a[i]][p] ? s[a[i]][p] - 'a' + 1 : 0`

40. ②处应填（　）

A. `tmp[ch][tail[ch]++] = a[i]`

B. `tmp[ch][++tail[ch]] = a[i]`

C. `tmp[ch][tail[ch]++] = i`

D. `tmp[ch][++tail[ch]] = i`

41. ③处应填（　）

A. `ans = (ans + 1ll * (p - 1) * tail[j] % mod) % mod`

B. `ans = (ans + 1ll * p * tail[j] % mod) % mod`

C. `ans = (ans + 1ll * (p + 1) * tail[j] % mod) % mod`

D. `ans = (ans + 1ll * (p + 2) * tail[j] % mod) % mod`

42. ④处应填（　）。

A. `ans = (ans + dfs(pos + 1, pos + tail[i], p + 1)) % mod`

B. `ans = (ans + dfs(pos + 1, pos + tail[i], p - 1)) % mod`

C. `ans = (ans + dfs(pos, pos + tail[i] - 1, p + 1)) % mod`

D. `ans = (ans + dfs(pos, pos + tail[i] - 1, p - 1)) % mod`

43. ⑤处应填（　）。

A. `dfs(0, n - 1, 0)`

B. `dfs(1, n, 0)`

C. `dfs(1, n, 1)`

D. `dfs(1, n - 1, 1)`

解析：关注公众号，回复“SCP2024”获得，考后一周内公开。