

YDSP 2024 · Junior 组初赛模拟考试

Copyright © 云斗学院 © 北斗学友教育科技有限公司

2024 年 9 月 14 日

本次考试的答题时间为两个小时，请各位选手自行掌握时间。

提交请访问 yundouxueyuan.com 参加对应的比赛，将最终答案以程序的形式提交至对应题目。

更进一步的提交细节，请参考对应题目里的说明。

最后，本次模拟考试为纯公益目的，不得用于任何未经授权的盈利活动。

* 祝考试顺利 *

2024 云斗学院软件能力认证第一轮

(YDSP - Junior) 入门级 C++ 语言试题

考生注意事项:

- 试题纸一共 14 页，满分 100 分。作答后请记录答案，并按要求提交至对应比赛处。
- 考试过程中不得使用任何电子设备或查阅任何书籍资料。

1 单选题 (每题 2 分, 共 30 分)

1.1 第 1 题

质因数分解是指将一个数字分成若干个质数的乘积。为了方便，如果质因数分解中出现了多个相同质因数，可以写成指数形式。下面 ▲ 是 2024 的质因数分解。

- A. $2^3 \times 11 \times 23$
- B. $2^4 \times 127$
- C. $2^2 \times 22 \times 23$
- D. $7 + 2017$

1.2 第 2 题

CCF NOI 系列比赛中，选手源代码长度不得超过 100 KB。现有四位选手各写了一份代码，长度依次是 2024 字节、41 MB、0.04 MB、114514 b。有 ▲ 个选手的代码没有超过长度限制。

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

1.3 第 3 题

在 C++ 中，0 开头的数字字面量是八进制的，如 `015==13`。在 C++ 中计算 `072 ^ 110` 的结果是 ▲。

- A. 84
- B. 78
- C. 38
- D. 0

1.4 第 4 题

若根的深度为 1，则拥有 2024 个结点，其中 921 个结点是叶子的二叉树深度最大是 ▲。

- A. 1102
- B. 1103
- C. 1104
- D. 不存在这样的二叉树

1.5 第 5 题

临近 CSP，小 A 想从去年云斗学院的 CSP-J 模拟赛和 CSP-S 模拟赛中各挑一道题进行练习，但是希望它们不属于同一知识点。已知 CSP-J 模拟赛的四题知识点依次为模拟、数学、模拟、DP，而 CSP-S 模拟赛的四题知识点依次为模拟、DP、高斯消元、并查集，请问小 A 有多少种选法？ ▲

- A. 10
- B. 13
- C. 15
- D. 16

1.6 第 6 题

计算后缀表达式 $4\ 3\ 4\ -\ -\ 6\ 2\ / \ 1\ * \ +$ 的结果是 ▲。

- A. 6
- B. 7
- C. 9
- D. 前三个选项都不对

1.7 第 7 题

执行如下代码片段后，*i 的值是 ▲。

```
1 vector<int> a = {10,20,30,40,50} ;  
2 auto i = a.end() - 3 ;
```

- A. 30
- B. 20
- C. 47
- D. 未定义行为

1.8 第 8 题

下列函数实现了翻转字符串 s 的第 $l \sim r$ 个字符的功能，例如 `range_rev("1234567",2,5)` 的结果是 1543267。空白处应填写 ▲。

```
1  string range_rev(string s, int l, int r) {  
2      for (int i = l - 1; i < (l + r) / 2; i++)  
3          swap(s[i], s[_____]);  
4      return s;  
5  }
```

- A. $l + r + 1 - i$
- B. $l + r - i$
- C. $l + r - 1 - i$
- D. $l + r - 2 - i$

1.9 第 9 题

一份文件仅含有英文小写字母，其中字母 a, b, c, \dots, z 出现次数依次为 $2^1, 2^2, \dots, 2^{26}$ ，若使用二进制哈夫曼编码方式，则字母 e 的编码长度是 ▲。

- A. 20
- B. 21
- C. 22
- D. 23

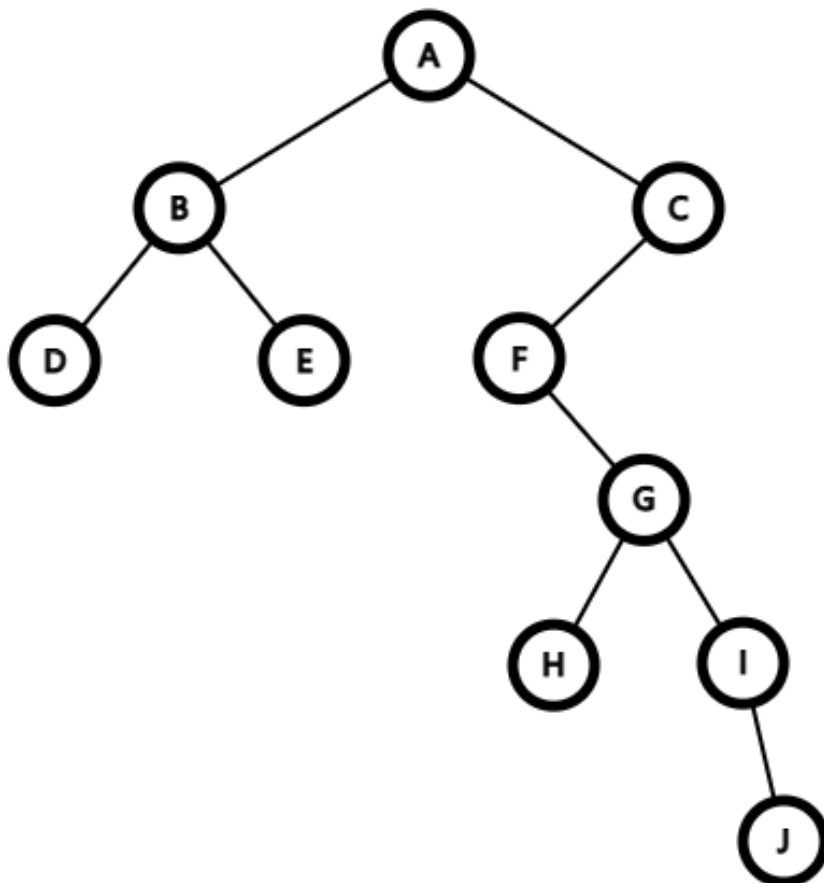
1.10 第 10 题

STL 是我们写代码的好帮手，下面关于 STL 的叙述中，正确的是 ▲。

- A. 若 a 是个 `list<int>`，那么 `a[3]` 可以求 a 的第四个元素。
- B. 若 a 是个 `queue<int>`，那么 `a.top()` 就是 a 的队首。
- C. `reserve` 函数可以用于反转一个序列。
- D. `sort` 函数填入适当的参数，就可以实现从大到小排序。

1.11 第 11 题

如图所示的二叉树中， $A \sim J$ 分别代表一个 $0 \sim 9$ 的自然数且不重复。已知该二叉树的前序遍历结果为 3192608574，则其中序遍历结果为 ▲ 。



- A. 9123547806
- B. 9215478063
- C. 9123605874
- D. 9123058746

1.12 第 12 题

4 个男生和 3 个女生排成一列，要求存在至少一个女生排在至少一个男生前面，有 ▲ 种排法。

- A. 8640
- B. 5039
- C. 4896
- D. 144

1.13 第 13 题

桌子上有背面朝上的 2, 3, 4, 5, 6, 7, 8 点扑克牌各一张, Bob 想要将这些牌从大到小排序, 但是不能直接翻看这些牌。

每次 Bob 可以选一些牌交给 Alice, 然后 Alice 可以指出这些牌中最大和最小的分别是哪张牌, 并向 Bob 收费一元。请问 Bob 至少要花多少元才能将这些牌从小到大排序? ▲

- A. 2
- B. 3
- C. 5
- D. 15

1.14 第 14 题

现有一个 3×3 的棋盘, 每个格子写了一个自然数。你要从左上角走到右下角, 每次只能向右或向下走一步, 求经过的所有格子数字和最大值。

一同学采用贪心算法, 每次从左上角开始, 从右边和下面的格子中选出数值更大的格子然后走一步 (若两个格子数值相等则随机选一个), 直到到达终点。要想让这个算法必然出错, 棋盘上这 9 个自然数中至多有 ▲ 个数为 0。

- A. 7
- B. 6
- C. 5
- D. 前三个选项都不对

1.15 第 15 题

一张简单无向图有 100 个结点和 200 条边, 下面三个情况中, 不可能的有 ▲ 个。

1. 有 20 个结点度数为 20。
2. 有 40 个结点度数为 10。
3. 有 90 个结点度数为 1。

- A. 0
- B. 1
- C. 2
- D. 3

2 阅读程序（无特殊说明时判断 1.5 分，选择 3 分。3 题共 40 分）

2.1 第 1 题（12 分）

```
1  #include<iostream>
2  #include<cstdio>
3  using namespace std;
4
5  char code[26];
6  string key;
7  string s;
8
9  void encode()
10 {
11     int len = s.length();
12     for(int i = 0; i < len; i++)
13         s[i] = code[((s[i] - 'a') + (key[i] - 'a')) % 26];
14 }
15
16 int main()
17 {
18     for(int i = 0; i < 26; i++) code[i] = i + 'a';
19     cin >> s >> key;
20     encode();
21     cout << s;
22     return 0;
23 }
```

程序正常运行的含义是：程序正常退出且输出的字符串长度与 s 相等。其他情况下为出错。假设输入的所有数据均为大写或小写字母，完成下列的判断题和选择题。

2.1.1 判断题

1. 如果输入的字符串 s 中有大写字母，那么程序一定会出错。
2. 如果输入的字符串 key 中有大写字母，那么程序可能会出错。
3. 正常情况下，输出结果一定全部为小写字母。
4. 为了保证程序正常运行，字符串 s 和字符串 key 必须长度相等。

2.1.2 选择题

5. 当输入的字符串依次为 yundou 和 ccfcsp 时, 输出结果为 ▲。
- A. awskfn
 - B. awsfhj
 - C. awsfgj
 - D. awskgn
6. 若输入的 key 字符串为 yundou, 输出的字符串为 rofgtw, 那么输入的 s 字符串为 ▲。
- A. tuskba
 - B. tusdfc
 - C. tuspkw
 - D. tusthu

2.2 第 2 题 (14 分)

阅读下面的程序, 回答问题。

```
1  #include <iostream>
2
3  const int mod = 29;
4  int n, t, m;
5  int prime[110], isprime[110], cnt;
6
7  void solve1(int n) {
8      for (int i = 1; i <= mod; i++) {
9          if (i * n % mod == 1) {
10             std::cout << i << std::endl;
11             return;
12         }
13     }
14     std::cout << -1 << std::endl;
15 }
16
17 void solve2(int n, int m) {
18     while (n != m) {
```



```

19     if (n < m) {
20         std::swap(n, m);
21     }
22     n -= m;
23 }
24 std::cout << n << std::endl;
25 }
26
27 void solve3(int n) {
28     for (int i = 2; i <= n; i++) {
29         if (!isprime[i]) {
30             prime[++cnt] = i;
31         }
32         for (int j = 1; j <= cnt; j++) {
33             if (i * prime[j] > n) {
34                 break;
35             }
36             isprime[i * prime[j]] = 1;
37             if (i % prime[j] == 0) {
38                 break;
39             }
40         }
41     }
42     std::cout << cnt << std::endl;
43 }
44
45 int main() {
46     std::cin >> t;
47     if (t == 1) {
48         std::cin >> n;
49         solve1(n);
50     } else if (t == 2) {
51         std::cin >> n >> m;
52         solve2(n, m);
53     } else if (t == 3) {
54         std::cin >> n;
55         solve3(n);

```

```
56     }
57     return 0;
58 }
```

程序满足输入 $1 \leq t \leq 3$, $1 \leq m, n \leq 100$, 且 n 不为 29 的倍数。

2.2.1 判断题

1. $t = 1$ 时, 有可能输出 -1 。
2. (2分)将第 18 行的 $n \neq m$ 改为 $n \&\& m$, 同时将第 24 行的 `std::cout << n << std::endl;` 改为 `std::cout << m << std::endl;` 之后, 结果不变。
3. 删除代码的第 37 行到 39 行后, 程序的输出依旧保持不变。

2.2.2 选择题

4. 当输入为 1 27 时, 输出为 ▲。
A. 18
B. 14
C. 27
D. 24
5. 当 $t = 2$ 时, 程序所求的结果为 ▲。
A. 最大公约数
B. 最小公倍数
C. 任意公约数
D. 任意公倍数
6. 当输入为 2 78 52 时, 输出为 ▲。
A. 2
B. 13
C. 3
D. 26

2.3 第 3 题 (14 分)

```
1  #include <cstring>
2  #include <iostream>
3  using namespace std;
4  #define int unsigned long long
5
6  int gcd(int a, int b) { return b ? gcd(b, a % b) : a; }
7
8  string s;
9  int num[4], cnt = 1, a, b, l[4];
10 bool f1, f2, f3, is_p;
11
12 int make_0(int l) {
13     int i = 1;
14     if (!l) return 1;
15     while (l--) i *= 10;
16     return i;
17 }
18
19 int make_9(int l) {
20     int i = 0;
21     while (l--) { i *= 10; i += 9; }
22     return i;
23 }
24
25 signed main() {
26     cin >> s;
27     for (int i = 0; i < s.size(); i++) {
28         if (s[i] == '.' ) {
29             if (s[i + 1] == '(' ) { f2 = 1; cnt = 3; }
30             else { f3 = 1; cnt = 2; }
31         }
32         if (s[i] == '(' ) { is_p = 1; cnt = 3; }
33         if (s[i] >= '0' && s[i] <= '9') {
34             num[cnt] *= 10, num[cnt] += (s[i] - '0');
35             l[cnt]++;
```

```

36     }
37 }
38
39 f1 = (!is_p);
40 if (f1) {
41     a = num[1] * make_0(l[2]) + num[2];
42     b = make_0(l[2]);
43 } else if (f2) {
44     a = num[3];
45     b = make_9(l[3]);
46     a = a + num[1] * b;
47 } else if (f3) {
48     a = num[2] * make_0(l[3]) + num[3] - num[2];
49     b = make_9(l[3]) * make_0(l[2]);
50     a = a + num[1] * b;
51 }
52
53 cout << a / gcd(a, b) << '/' << b / gcd(a, b);
54 return 0;
55 }

```

输入数据满足 $1 \leq |s| \leq 10^3$ 。试完成以下判断题和单选题：

2.3.1 判断题

1. 删去第 14 行对输出不造成影响。
2. (2 分) 输入 2024.06(191620) 时，输出为 707713249 349650。
3. 输入 1 会导致程序出现运行错误。

2.3.2 选择题

4. (2 分) 输入 2(1) 时，输出为 ▲。
 - A. 0/0
 - B. 21/1
 - C. 2/1
 - D. 无输出

5. 分别记第 40 ~ 42 行、第 43 ~ 46 行、第 47 ~ 50 行分别为代码【块 1】、【块 2】和【块 3】。
当输入为 .2(2) 时，将调用 ▲ 。

A. 【块 1】

B. 【块 2】

C. 【块 3】

D. 【块 2】，【块 3】

6. (4 分) 当输入为 .21629(629) 时，a 与 b 的值分别为 ▲ 。

A. 146, 675

B. 146000, 675000

C. 21608, 99900

D. 21608000, 99900000

3 完善程序 (2 题, 每空 3 分, 共 30 分)

3.1 相等最大权值定量变换

给定数列 $\{b_n\}$ 。数列 $\{a_n\}$ 初始各项均为 1。

每次可以选择正整数 i, x ($1 \leq i \leq n$), 将 a_i 变为 $a_i + \lfloor \frac{a_i}{x} \rfloor$ 。其中 $\lfloor z \rfloor$ 表示对 z 向下取整。

最多可以进行 k 次操作, 操作后若 $a_i = b_i$ 可以得到得分 c_i , 求出最大得分。

已知: $1 \leq n \leq 10^3$, $1 \leq k \leq 10^6$, $1 \leq b_i \leq 10^3$, $1 \leq c_i \leq 10^6$ 。

提示: 考虑预处理出对于所有 x , $1 \rightarrow x$ 所需的最小操作数 f_x 。不难发现, 本题目的本质是一个背包问题, 直接 dp 即可。

试补全以下程序。

```
1  #include <bits/stdc++.h>
2
3  #define N 1000010
4  #define M 1000
5  #define INF 1000010
6  #define ll long long
7
8  using namespace std;
9
10 inline ll rd();
11 ll n, k, cnt, sum, b[N], c[N], f[N], dp[N];
12 ll mx = 1;
13
14 void init(){
15     for(int i = 1 ; i <= M ; i++)
16         f[i] = INF;
17     f[1] = 0;
18     for(int i = 1 ; i <= M ; i++)
19         for(int j = 1 ; j <= i ; j++)
20             /* Blank 1 */;
21     mx = max(mx, f[i + i / j]);
22 }
23
24 int main(){
25     init();
26     n = rd(), k = rd();
```

```

27     for(int i = 1 ; i <= n ; i++){
28         b[i] = rd();
29         /* Blank 2 */;
30     }
31     for(int i = 1 ; i <= n ; i++)
32         c[i] = rd(), sum += c[i];
33     if(/* Blank 3 */){
34         printf("%lld\n", sum);
35         return 0;
36     }
37     memset(dp, 0, sizeof(dp));
38     for(int i = 1 ; i <= n ; i++)
39         /* Blank 4 */ ;
40         /* Blank 5 */
41     printf("%lld\n", dp[k]);
42     return 0;
43 }
44 inline ll rd(){
45     char c;
46     bool flag = 0;
47     while((c = getchar()) < '0' || c > '9')
48         if(c == '-') flag = 1;
49     ll res = c - '0';
50     while((c = getchar()) >= '0' && c <= '9')
51         res = (res << 3) + (res << 1) + c - '0';
52     return flag ? -res : res;
53 }

```

1. 第 /* Blank 1 */ 空应该填 ▲。

- A. $f[i + i / j] = \min(f[i + i / j], f[i])$
- B. $f[i + i / j] = \min(f[i + i / j], f[i] + 1)$
- C. $f[i + (i+1) / j] = \min(f[i + (i+1) / j], f[i])$
- D. $f[i + (i+1) / j] = \min(f[i + (i+1) / j], f[i] + 1)$

2. 第 /* Blank 2 */ 空应该填 ▲。

- A. $sum += b[i]$

- B. `sum += f[b[i]]`
- C. `cnt += b[i]`
- D. `cnt += f[b[i]]`
3. 第 /* Blank 3 */ 空应该填 ▲ 。
- A. `k >= cnt`
- B. `k >= cnt * n`
- C. `k >= sum`
- D. `k >= sum * n`
4. 第 /* Blank 4 */ 空应该填 ▲ 。
- A. `for(int j = k ; j >= f[b[i]] ; j--)`
- B. `for(int j = k ; j ; j--)`
- C. `for(int j = f[b[i]] ; j <= k ; j++)`
- D. `for(int j = 1 ; j <= k ; j++)`
5. 第 /* Blank 5 */ 空应该填 ▲ 。
- A. `dp[j] = min(dp[j - f[b[i]]], dp[j] + c[i]);`
- B. `dp[j] = min(dp[j - f[b[i]]] + c[i], dp[j]);`
- C. `dp[j] = max(dp[j - f[b[i]]], dp[j] + c[i]);`
- D. `dp[j] = max(dp[j - f[b[i]]] + c[i], dp[j]);`

3.2 最大边权最短路

给定一张 n 个结点 m 条边的无向有权图，求 s 到 t 的所有路径中，**最大边权的最小值**。

保证 $1 \leq n, m \leq 10^5$ ，且保证边权均为不超过 10^6 的正整数。

下面的程序使用二分答案 + DFS 解决该问题，时间复杂度 $O((n + m) \log w)$ ，尝试补全代码。

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  int n, m, s, t;
5  bool vis[100005];
6  struct edge {
7      int to, dis;

```



```

8   };
9   vector<edge> g[100005];
10
11  bool check(int maxdis, int cur) {
12      if (cur == t) return true;
13      if (/* Blank 1 */) return false;
14      vis[cur] = true;
15      for (int i = 0; i < g[cur].size(); i++)
16          if (g[cur][i].dis <= maxdis && /* Blank 2 */)
17              return true;
18      /* Blank 3 */
19  }
20
21  int main() {
22      cin >> n >> m >> s >> t;
23      for (int i = 1; i <= m; i++) {
24          int u, v, w;
25          cin >> u >> v >> w;
26          g[u].push_back({v, w});
27          g[v].push_back({u, w});
28      }
29
30      int l = 0, r = /* Blank 4 */;
31      while (l < r) {
32          int mid = (l + r) / 2;
33          memset(vis, 0, sizeof vis);
34          if (check(mid, s))
35              r = mid;
36          else
37              l = /* Blank 5 */;
38      }
39      cout << l;
40      return 0;
41  }

```

1. 第 /* Blank 1 */ 空应该填 ▲ 。
 - A. vis[cur]
 - B. cur > maxdis
 - C. maxdis <= 0
 - D. g[cur].size() == 0

2. 第 /* Blank 2 */ 空应该填 ▲ 。
 - A. check(g[cur][i].dis, g[cur][i].to)
 - B. check(min(maxdis, g[cur][i].dis), i)
 - C. check(maxdis+g[cur][i].dis, g[cur][i].to)
 - D. check(maxdis, g[cur][i].to)

3. 第 /* Blank 3 */ 空应该填 ▲ 。
 - A. vis[cur]=false; return false;
 - B. vis[cur]=false; return true;
 - C. return false;
 - D. return vis[cur];

4. 第 /* Blank 4 */ 空应该填 ▲ 。
 - A. n
 - B. m
 - C. 10000000000
 - D. 1000000

5. 第 /* Blank 5 */ 空应该填 ▲ 。
 - A. mid
 - B. mid + (l + r) % 2
 - C. mid - 1
 - D. l + 1