2023 CCF 非专业级别软件能力认证第一轮

(CSP-J1) 入门级 C++语言试题

认证时间：2023 年 9 月 16 日 09:30~11:30

考生注意事项：

● 试题纸共有 10 页，答题纸共有 1 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 2 分， 共计 30 分；每题有且有一个正确选项）

1．在 C++中，下面哪个关键字用于声明一个变量，其值不能被修改？（ ）

A．unsigned

B．const

C．static

D．mutable

2．八进制数 12345678_8 和 07654321_8 的和为()。

A．222222218

B．211111118

C．221111118

D．222222118

3．阅读下述代码，请问修改 data 的 value 成员以存储 3.14，正确的方式是()

```
union Data {
    int num;
    float value;
    char symbol;
};
union Data data;
```

A．data.value = 3.14;

B．value.data = 3.14;

C．data->value = 3.14;

D．value->data = 3.14;

4．假设有一个链表的节点定义如下：

```
struct Node {
    int data;
    Node* next;
};
```

现在有一个指向链表头部的指针：Node* head。如果想要在链表中插入一个新的节点，其成员 data 的值为 42，并使新节点成为链表的第一个节点，下面哪个操作是正确的？（　）

A．Node* newNode = new Node; newNode->data = 42; newNode->next = head; head = newNode;

B．Node* newNode = new Node; head->data = 42; newNode->next = head; head = newNode;

C．Node* newNode = new Node; newNode->data = 42; head->next = newNode;

D．Node* newNode = new Node; newNode->data = 42; newNode->next = head;

5．根节点的高度为 1，一棵拥有 2023 个节点的三叉树高度至少为()

A．6

B．7

C．8

D．9

6．小明在某一天中依次有七个空闲时间段，他想要选出至少一个空闲时间段来练习唱歌，但他希望任意两个练习的时间段之间都有至少两个空闲的时间段让他休息。则小明一共有(　) 种选择时间段的方案

A．31

B．18

C．21

D．33

7．以下关于高精度运算的说法错误的是(　)。

A．高精度计算主要是用来处理大整数或需要保留多位小数的运算。

B．大整数除以小整数的处理的步骤可以是，将被除数和除数对齐，从左到右逐位尝试将除数乘以某个数，通过减法得到新的被除数，并累加商。

C．高精度乘法的运算时间只与参与运算的两个整数中长度较长者的位数有关。

D．高精度加法运算的关键在于逐位相加并处理进位。

8. 后缀表达式“6 2 3 + - 3 8 2 / + * 2 3 +”对应的中缀表达式是(　　)。

A. ((6 - (2 + 3)) * (3 + 8 / 2)) ^ 2 + 3

B. 6 – 2 + 3 * 3 + 8 / 2 ^ 2 + 3

C. (6 - (2 + 3)) * ((3 + 8 / 2) ^ 2) + 3

D. 6 - ((2 + 3) * (3 + 8 / 2)) ^ 2 + 3

9. 数 101010_2 和 166_8 的和为(　　)

A. 10110000_2

B. 236_8

C. 158_{10}

D. $A0_{16}$

10. 假设有一组字符{a,b,c,d,e,f}，对应的频率分别为 5%、9%、12%、13%、16%、45%。请问以下哪个选项是字符 a,b,c,d,e,f 分别对应的一组哈夫曼编码?(　　)

A. 1111，1110，101，100，110，0

B. 1010，1001，1000，011，010，00

C. 000，001，010，011，10，11

D. 1010，1011，110，111，00，01

11. 给定一棵二叉树,其前序遍历结果为: ABDECFG,中序遍历结果为: DEBACFG。请问这棵树的正确后序遍历结果是什么?(　　)

A. EDBFGCA

B. EDBGCFA

C. DEBGFCA

D. DBEGFCA

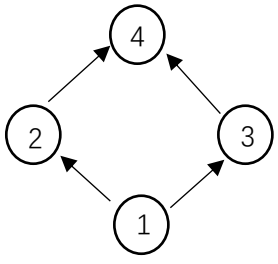
12. 考虑一个有向无环图，该图包含 4 条有向边:(1,2)，(1,3)，(2,4)和(3,4)。以下哪个选项是这个有向无环图的一个有效的拓扑排序?(　　)

A.4, 2, 3, 1

B.1, 2, 3, 4

C.1, 2, 4, 3

D.2, 1, 3, 4



13.在计算机中，以下哪个选项描述的数据存储容量最小?(　　)

A. 字节(byte)

B. 比特(bit)

C. 字(word)

D. 千字节(kilobyte)

14.一个班级有 10 个男生和 12 个女生。如果要选出一个 3 人的小组，并且小组中必须至少包含 1 个女生，那么有多少种可能的组合?(　　)

A. 1420

B. 1770

C. 1540

D. 2200

15.以下哪个不是操作系统?(　　)

A. Linux

B. Windows

C. Android

D. HTML

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

(1)

```
01 #include <iostream>
02 #include <cmath>
03 using namespace std;
04
05 double f(double a, double b, double c) {
06     double s = (a + b + c) / 2;
07     return sqrt(s * (s - a) * (s - b) * (s - c));
08 }
09
10 int main() {
11     cout.flags(ios::flxed);
12     cout.precision(4);
13
14     int a, b, c;
```

```
15      cin >> a >> b >> c;
16      cout << f(a, b, c) << endl;
17      return 0;
18  }
```

假设输入的所有数都为不超过 1000 的正整数，完成下面的判断题和单选题判断题：

- 判断题

16. (2 分) 当输入为“2 2 2”时，输出为“1.7321”。( )
17. (2 分) 将第 7 行中的“(s - b)*(s - c)”改为“(s - c)*(s - b)” 不会影响程序运行的结果。( )
18. (2 分) 程序总是输出四位小数。( )

- 单选题

19. 当输入为“3 4 5”时，输出为( )。
A. “6.0000” B. “12.0000” C. “24.0000” D. “30.0000”

20. 当输入为“5 12 13”时，输出为( )
A. “24.0000” B. “30.0000” C. “60.0000” D. “120.0000”

(2)

```
01 #include <iostream>
02 #include <vector>
03 #include <algorithm>
04 using namespace std;
05
06 int f(string x, string y) {
07     int m = x.size();
08     int n = y.size();
09     vector<vector<int>> v(m+1,vector<int>(n+1,0));
10     for(int i = 1; i <= m; i++) {
11         for(int j = 1; j <= n; j++) {
12             if(x[i-1] == y[j-1]) {
13                 v[i][j] = v[i-1][j-1] + 1;
14             } else {
15                 v[i][j] = max(v[i-1][j], v[i][j-1]);
16             }
17         }
18     }
19     return v[m][n];
20 }
```

```
21
22 bool g(string x, string y) {
23     if(x.size() != y.size()) {
24         return false;
25     }
26     return f(x + x, y) == y.size();
27 }
28
29 int main() {
30     string x, y;
31     cin >> x >> y;
32     cout << g(x, y) << endl;
33     return 0;
34 }
```

● 判断题

21.f 函数的返回值小于等于 min(n,m)。( )
22.f 函数的返回值等于两个输入字符串的最长公共子串的长度。( )
23.当输入两个完全相同的字符串时，g 函数的返回值总是 true.( )

● 单选题

24.将第 19 行中的“v[m][n]” 替换为 “v[n][m]”，那么该程序( )。
A. 行为不变 B. 只会改变输出 C. 一定非正常退出 D. 可能非正常退出

25.当输入为“csp-j p-jcs”时，输出为( )。
A. “0” B. “1” C. “T” D. “F”

26.当输入为“csppsc spsccp”时，输出为( )。
A. “T” B. “F” C. “0” D. “1”

(3)

```
01 #include <iostream>
02 #include <cmath>
03 using namespace std;
04
05 int solve1(int n) {
06     return n * n;
07 }
08
09 int solve2(int n) {
```

```
10      int sum = 0;
11      for (int i = 1; i <= sqrt(n); i++) {
12          if(n % i == 0) {
13              if(n/i == i) {
14                  sum += i*i;
15              } else {
16                  sum += i*i + (n/i)*(n/i);
17             }
18          }
19      }
20      return sum;
21  }
22
23  int main() {
24      int n;
25      cin >> n;
26      cout<<solve2(solve1(n))<<" "<<solve1(solve2(n))<< endl;
27      return 0;
28  }
```

假设输入的 n 是绝对值不超过 1000 的整数，完成下面的判断题和单选题：

- 判断题

27.如果输入的 n 为正整数，solve2 函数的作用是计算 n 所有的因子的平方和。(　　)

28.第 13-14 行的作用是避免 n 的平方根因子(或 n/i)进入第 16 行而被计算两次。(　　)

29.如果输入的 n 为质数，solve2(n)的返回值为 n^2+1。(　　)

- 单选题

30.(4 分)如果输入的 n 为质数 p 的平方，那么 solve2(n)的返回值为(　)。

A. $p^2 + p + 1$　B. $n^2 + n + 1$　C. $n^2 + 1$　D. $p^4 + 2p^2 + 1$

31.当输入为正整数时，第一项减去第二项的差值一定(　　)。

A. 大于等于 0　　B.大于等于 0 且不一定大于 0

C. 小于 0　　D. 小于等于 0 且不一定小于 0

32.当输入为“5”时，输出为(　　)。

A. “651 625”　B. “650 729”　C. “651 676”　D.”652 625”

三、完善程序（单选题，每小题 3 分，共计 30 分）

(1)（寻找被移除的元素）问题：原有长度为 n+1、公差为 1 的等差升序数列；将数列输入到程序的数组时移除了一个元素，导致长度为 n 的升序数组可能不再连续，除非被移除的是第一个或最后一个元素。 需要在数组不连续时， 找出被移除的元素。

试补全程序。

```
01 #include <iostream>
02 #include <vector>
03
04 using namespace std;
05
06 int find_missing(vector<int>& nums) {
07     int left = 0, right = nums.size() - 1;
08     while (left < right) {
09         int mid = left + (right - left) / 2;
10         if (nums[mid] == mid + ①) {
11             ②
12         } else {
13             ③
14         }
15      }
16      return ④
17 }
18
29 int main() {
20     int n;
21     cin >> n;
22     vector<int> nums(n);
23     for(int i = 0; i < n; i++)cin >> nums[i];
24     int missing_number = find_missing(nums);
25     if (missing_number == ⑤) {
26         cout << "Sequence is consecutive" << endl;
27     } else {
28         cout << "Missing number is " << missing_number << endl;
29     }
30     return 0;
31 }
```

33. ①处应该填(　　)

A. 1　　B. nums[0]　　C. right　　D. left

34. ②处应该填(　)

A. left = mid + 1　　B. right = mid – 1

C. right = mid　　D. left = mid

35. ③处应该填(　)

A. left = mid + 1　　B. right = mid – 1

C. right = mid　　D. left = mid

36. ④处应该填(　)

A. left = nums[0]　　B. right + nums[0]

C. mid + nums[0]　　D. right + 1

37. ⑤处应该填(　)

A. nums[0]+n　B. nums[0]+n-1　C. nums[0]+n+1　D. nums[n-1]

(2)(编辑距离)给定两个字符串，每次操作可以选择删除(Delete)、插入(insert)替换(Replace)个字符，求将第一个字符串转换为第二个字符串所需要的最少操作次数。

试补全动态规划算法。

```
01 #include <iostream>
02 #include <string>
03 #include <vector>
04 using namespace std;
05
06 int min(int x, int y, int z) {
07     return min(min(x, y), z);
08 }
09
10 int edit_dist_dp(string str1, string str2) {
11     int m = str1.length();
12     int n = str2.length();
13     vector<vector<int>> dp(m + 1, vector<int>(n + 1));
14
15     for (int i = 0; i <= m; i++) {
16         for (int j = 0; j <= n; j++) {
17             if (i == 0)
18                 dp[i][j] = ①;
19             else if (j == 0)
20                 dp[i][j] = ②;
21             else if (③)
22                 dp[i][j] = ④;
23             else
```

```
24                dp[i][j]=1+min(dp[i][j - 1],dp[i - 1][j], ⑤);
25            }
26         }
27     return dp[m][n];
28 }
29
30 int main() {
31     string str1, str2;
32     cin >> str1 >> str2;
33     cout << "Mininum number of operation:"
34          << edit_dist_dp(str1, str2) << endl;
35     return 0;
35 }
```

38. ①处应该填( )

A. j　　B. i　　C. m　　D. n

39. ②处应该填( )

A. j　　B. i　　C. m　　D. n

40. ③处应该填( )

A. str1[i – 1] == str2[j – 1]　　B. str1[i] == str2[j]

C. str1[i – 1] != str2[j – 1]　　D. str1[i] != str2[j]

41. ④处应该填( )

A. dp[i – 1][j – 1] + 1　　B. dp[i – 1][j – 1]

C. dp[i – 1][j]　　D. dp[i][j – 1]

42. ⑤处应该填( )

A. dp[i][j] + 1　　B. dp[i – 1][j – 1] + 1

C. dp[i – 1][j – 1]　　D. dp[i][j]