2022年CCF非专业级别软件能力认证第一轮 （CSP-J）入门级C++语言试题

题目总数：20　总分数：100

一、单项选择题

第 1 题　单选题

以下哪种功能没有涉及 C++语言的面向对象特性支持：（ ）

A. C++中调用 printf 函数

B. C++中调用用户定义的类成员函数

C. C++中构造一个 class 或 struct

D. C++中构造来源于同一基类的多个派生类

答案　A

解析　printf 是继承自C 的，C 是纯面向过程的

第 2 题　单选题

有 6 个元素，按照 6、5、4、3、2、1 的顺序进入栈 S，请问下列哪个出栈序列是非法的（ ）。

A. 5 4 3 6 1 2

B. 4 5 3 1 2 6

C. 3 4 6 5 2 1

D. 2 3 4 1 5 6

答案　C

解析　栈先进后出，C选项3，4出栈，说明此时6，5都在栈中，这时不可能6先出。

第 3 题　单选题

运行以下代码片段的行为是（ ）。

青少年考级竞赛题库--在线刷题/真题模拟/不限科目/顺利通过-https://tiku.scratchor.com/

```
int x = 101;
int y = 201;
int *p = &x;
int *q = &y;
p = q;
```

A. 将 x 的值赋为 201
B. 将 y 的值赋为 101
C. 将 q 指向 x 的地址
D. 将 p 指向 y 的地址

答案 D

解析 考察指针和地址。第三行p指向了x，第四行q指向了y，最后p改指y。

第 4 题 单选题

链表和数组的区别包括（ ）。

A. 数组不能排序，链表可以
B. 链表比数组能存储更多的信息
C. 数组大小固定，链表大小可动态调整
D. 以上均正确

答案 C

解析 这个题其实不太严谨，但C相比其他更正确。

第 5 题 单选题

对假设栈 S 和队列 Q 的初始状态为空。存在 e1~e6 六个互不相同的数据，每个数据按照进栈 S、出栈 S、进队列 Q、出队列 Q 的顺序操作，不同数据间的操作可能会交错。已知栈 S 中依次有数据 e1、e2、e3、e4、e5 和 e6 进栈，队列 Q 依次有数据 e2、e4、e3、e6、e5 和 e1 出队列。则栈 S 的容量至少是（ ）个数据。

A. 2
B. 3
C. 4
D. 6

答案 B

解析 出栈、进队、出队连着的，因此直接视为出栈就行了。按照出栈序列，至少3个。

青少年考级竞赛题库--在线刷题/真题模拟/不限科目/顺利通过-https://tiku.scratchor.com/

第 6 题　单选题

对表达式 a+(b-c)*d 的前缀表达式为（ ），其中+、-、*是运算符。

A. *+a-bcd

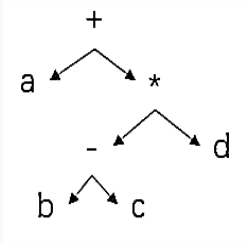
B. +a*-bcd

C. abc-d*+

D. abc-+d

答案　B

解析　如图所示画出表达式树，而后先序遍历即可。



第 7 题　单选题

假设字母表 {a, b, c, d, e} 在字符串出现的频率分别为 10%, 15%, 30%, 16%, 29%。若使用哈夫曼编码方式对字母进行不定长的二进制编码，字母 d 的编码长度为（ ）位。

A. 1

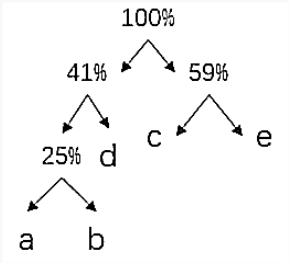
B. 2

C. 2或3

D. 3

答案　B

解析　如图所示画出哈夫曼树，看 d 所在的层数即可。



第 8 题　单选题

一棵有 n 个结点的完全二叉树用数组进行存储与表示，已知根结点存储在数组的第 1 个位置。若存储在数组第 9 个位置的结点存在兄弟结点和两个子结点，则它的兄弟结点和右子结点的位置分别是（ ）。

A. 8,18

B. 10,18

C. 8,19

D. 10,19

青少年考级竞赛题库--在线刷题/真题模拟/不限科目/顺利通过-https://tiku.scratchor.com/

答案 C

解析 考察完全二叉树的线性数组表示法。根节点i的左右孩子分别是2i和2i+1。也可以直接画出二叉树观察。

第 9 题 单选题

考虑由 N 个顶点构成的有向连通图，采用邻接矩阵的数据结构表示时，该矩阵中至少存在（ ）个非零元素。

A. N-1

B. N

C. N+1

D. N^2

答案 A

解析 N个节点连通，至少需要N-1条边。

第 10 题 单选题

以下对数据结构的表述不恰当的一项为：（ ）。

A. 图的深度优先遍历算法常使用的数据结构为栈。

B. 栈的访问原则为后进先出，队列的访问原则是先进先出。

C. 队列常常被用于广度优先搜索算法。

D. 栈与队列存在本质不同，无法用栈实现队列。

答案 D

解析 用两个栈“栈底相连”模拟一个队列是常见的操作，虽然没人这么做就是了。

第 11 题 单选题

以下哪组操作能完成在双向循环链表结点 p 之后插入结点 s 的效果（其中，next 域为结点的直接后继，prev 域为结点的直接前驱）：（ ）。

A. p->next->prev=s; s->prev=p; p->next=s; s->next=p->next;

B. p->next->prev=s; p->next=s; s->prev=p; s->next=p->next;

C. s->prev=p; s->next=p->next; p->next=s; p->next->prev=s;

D. s->next=p->next; p->next->prev=s; s->prev=p; p->next=s;

答案 D

解析 观察选项可知，四个选项为这四个语句的不同排列：

s->prev=p;

s->next=p->next;

青少年考级竞赛题库--在线刷题/真题模拟/不限科目/顺利通过-https://tiku.scratchor.com/

p->next=s;
p->next->prev=s;
这里发生变化，又可能给其它量赋值的就是p->next。
使p->next发生变化的语句为：p->next=s;
而s->next=p->next;与p->next->prev=s;
中用到的都应该是变化前的p->next，指向的是原来p的下一个结点。
所以p->next=s;应该放在最后，选D。

第 12 题　单选题

以下排序算法的常见实现中，哪个选项的说法是错误的：（ ）。

A. 冒泡排序算法是稳定的
B. 简单选择排序是稳定的
C. 简单插入排序是稳定的
D. 归并排序算法是稳定的

答案　B

解析　考察排序的稳定性。选择排序是不稳定的，冒泡、插入、归并都是稳定的。

第 13 题　单选题

八进制数 32.1 对应的十进制数是（ ）。

A. 24.125
B. 24.250
C. 26.125
D. 26.250

答案　C

解析　考察 N 进制转 10 进制。整数部分从低到高分别乘以 N^0, N^1, N^2, ...；小数部分自然是 N^-1，N^-2，...。按此规律计算即可。

第 14 题　单选题

一个字符串中任意个连续的字符组成的子序列称为该字符串的子串，则字符串 abcab 有（ ）个内容互不相同的子串。

A. 12
B. 13
C. 14
D. 15

答案　B

青少年考级竞赛题库--在线刷题/真题模拟/不限科目/顺利通过-https://tiku.scratchor.com/

解析

abcab的不相同子串有：
长为1的子串：a,b,c
长为2的子串：ab,bc,ca
长为3的子串：abc,bca,cab
长为4的子串：abca,bcab
长为5的子串：abcab
再加上一个空串（任意个）
共有13个。

第 15 题　单选题

以下对递归方法的描述中，正确的是：（ ）

A. 递归是允许使用多组参数调用函数的编程技术
B. 递归是通过调用自身来求解问题的编程技术
C. 递归是面向对象和数据而不是功能和逻辑的编程语言模型
D. 递归是将用某种高级语言转换为机器代码的编程技术

答案

B

解析

选项A：不清楚什么叫“多组参数调用函数”，任意一个带参函数，都可以用多组参数来调用。
选项B：论述正确。
选项C：面向对象编程（或者说“类”）是面向对象和数据而不是功能和逻辑的编程语言模型。
选项D：编译是将用某种高级语言转换为机器代码的编程技术

二、阅读程序

第 16 - 21 题　组合题

```
01 #include <iostream>
02
03 using namespace std;
04
05 int main()
06 {
07 unsigned short x, y;
08 cin >> x >> y;
09 x = (x | x << 2) & 0x33;
10 x = (x | x << 1) & 0x55;
11 y = (y | y << 2) & 0x33;
12 y = (y | y << 1) & 0x55;
13 unsigned short z = x | y << 1;
```

青少年考级竞赛题库--在线刷题/真题模拟/不限科目/顺利通过-https://tiku.scratchor.com/

```
14 cout << z << endl;
15 return 0;
16 }
```

假设输入的 x、y 均是不超过 15 的自然数，完成下面的判断题和单选题：

第 16 题　判断题

删去第 7 行与第 13 行的 unsigned，程序行为不变。（ ）

A. 正确

B. 错误

答案　A

第 17 题　判断题

将第 7 行与第 13 行的 short 均改为 char，程序行为不变。（ ）

A. 正确

B. 错误

答案　B

第 18 题　判断题

程序总是输出一个整数“0”。（ ）

A. 正确

B. 错误

答案　B

第 19 题　判断题

当输入为“2 2”时，输出为“10”。（ ）

A. 正确

B. 错误

答案　B

第 20 题　判断题

当输入为“2 2”时，输出为“59”。（ ）

A. 正确

B. 错误

答案　B

第 21 题　单选题

当输入为“13 8”时，输出为（ ）。

青少年考级竞赛题库--在线刷题/真题模拟/不限科目/顺利通过-https://tiku.scratchor.com/

A. “0”
B. “209”
C. “197”
D. “226”

答案 B

解析
考察位运算及其优先级、十六进制。易错点包括左移运算优先于按位或运算。
1）x，y 不超过15，即 1111，程序中的各种运算不会让他们超出 16 位，因此是否使用符号位做数据位并不影响。
2）short 是 16 位，char 只有 8 位，经过操作 8 位会越界。
3）事实上，下一个题目的“2 2”带入就不是。如果最后是 x & y << 1，那么在 x 和 y 相等的情况下确实输出“0”。
4）代入计算可以得到，应为 12。
5）同上一题
6）纯计算题，带入小心计算即可。

第 17 - 22 题　组合题

```
01 #include <algorithm>
02 #include <iostream>
03 #include <limits>
04
05 using namespace std;
06
07 const int MAXN = 105;
08 const int MAXK = 105;
09
10 int h[MAXN][MAXK];
11
12 int f(int n, int m)
13 {
14 if (m == 1) return n;
15 if (n == 0) return 0;
16
17 int ret = numeric_limits<int>::max();
18 for (int i = 1; i <= n; i++)
19 ret = min(ret, max(f(n - i, m), f(i - 1, m - 1)) + 1);
20 return ret;
21 }
22
23 int g(int n, int m)
```

青少年考级竞赛题库--在线刷题/真题模拟/不限科目/顺利通过-https://tiku.scratchor.com/

```
24 {
25 for (int i = 1; i <= n; i++)
26 h[i][1] = i;
27 for (int j = 1; j <= m; j++)
28 h[0][j] = 0;
29
30 for (int i = 1; i <= n; i++) {
31 for (int j = 2; j <= m; j++) {
32 h[i][j] = numeric_limits<int>::max();
33 for (int k = 1; k <= i; k++)
34 h[i][j] = min(
35 h[i][j],
36 max(h[i - k][j], h[k - 1][j - 1]) + 1);
37 }
38 }
39
40 return h[n][m];
41 }
42
43 int main()
44 {
45 int n, m;
46 cin >> n >> m;
47 cout << f(n, m) << endl << g(n, m) << endl;
48 return 0;
49 }
```

假设输入的 n、m 均是不超过 100 的正整数，完成下面的判断题和单选题：

第 17 题　判断题

当输入为“7 3”时，第 19 行用来取最小值的 min 函数执行了 449 次。（ ）

A. 正确

B. 错误

答案　B

第 18 题　判断题

输出的两行整数总是相同的。（ ）

A. 正确

B. 错误

答案　A

第 19 题　判断题

当 m 为 1 时，输出的第一行总为 n。（ ）

A. 正确

B. 错误

答案 A

第 20 题 单选题

算法 g(n,m)最为准确的时间复杂度分析结果为（ ）。

A. 0（n^3/2m)

B. 0(nm)

C. 0(n^2m)

D. 0(nm^2)

答案 C

第 21 题 单选题

当输入为“20 2”时，输出的第一行为（ ）。

A. “4”

B. “5”

C. “6”

D. “20”

答案 C

第 22 题 单选题

当输入为“100 100”时，输出的第一行为（ ）。

A. “6”

B. “7”

C. “8”

D. “9”

答案 C

解析

1）考察调用栈原理，J组最难算的一题。设 f[i][j] 为 n = i，m = j 时的min次数。根据递归出口有，f[0][1~m]=0，f[i][1]=i。根据递归，有转移方程：

$$f[n][m] = \sum_{i=1}^{n} f[n-i][m] + f[i-1][m-1]$$

计算后答案为 448 次。

2）两个函数功能是相同的，f 是递归写法，g 是 dp 写法。

3）直接走出口了。

4）与基础排序算法类似的复杂度分析，加上一层 m，综合时间复杂度为 O(1/2n^2m)。

青少年考级竞赛题库--在线刷题/真题模拟/不限科目/顺利通过-https://tiku.scratchor.com/

5）第一行第二行输出相同，直接带入 dp 填表即可。

6）本题的背景实际上是高楼扔鸡蛋模型，即 n 层楼，m 个鸡蛋，求最高的鸡蛋扔下不会碎的楼层。在鸡蛋无限时，可以直接二分，ans = logn下取整 + 1。

第 18 - 24 题　组合题

```
01 #include <iostream>
02
03 using namespace std;
04
05 int n, k;
06
07 int solve1()
08 {
09 int l = 0, r = n;
10 while (l <= r) {
11 int mid = (l + r) / 2;
12 if (mid * mid <= n) l = mid + 1;
13 else r = mid - 1;
14 }
15 return l - 1;
16 }
17
18 double solve2(double x)
19 {
20 if (x == 0) return x;
21 for (int i = 0; i < k; i++)
22 x = (x + n / x) / 2;
23 return x;
24 }
25
26 int main()
27 {
28 cin >> n >> k;
29 double ans = solve2(solve1());
30 cout << ans << ' ' << (ans * ans == n) << endl;
31 return 0;
32 }
```

假设 int 为 32 位有符号整数类型，输入的 n 是不超过 47000 的自然数、k 是不超过 int表示范围的自然数，完成下面的判断题和单选题：

第 18 题　判断题

该算法最准确的时间复杂度分析结果为0(log n + k)。（ ）

A.　正确

B.　错误

答案 A

第 19 题　判断题

当输入为“9801 1”时，输出的第一个数为“99”。（ ）

A. 正确

B. 错误

答案 A

第 20 题　判断题

对于任意输入的 n，随着所输入 k 的增大，输出的第二个数会变成“1”。（ ）

A. 正确

B. 错误

答案 B

第 21 题　判断题

该程序有存在缺陷。当输入的 n 过大时，第 12 行的乘法有可能溢出，因此应当将mid 强制转换为 64 位整数再计算。（ ）

A. 正确

B. 错误

答案 B

第 22 题　单选题

当输入为“2 1”时，输出的第一个数最接近（ ）。

A. 1

B. 1.414

C. 1.5

D. 2

答案 C

第 23 题　单选题

当输入为“3 10”时，输出的第一个数最接近（ ）。

A. 1.7

B. 1.732

C. 1.75

D. 2

B

青少年考级竞赛题库--在线刷题/真题模拟/不限科目/顺利通过-https://tiku.scratchor.com/

答案

第 24 题　单选题

当输入为“256 11”时，输出的第一个数（ ）。

A. 等于 16

B. 接近但小于 16

C. 接近但大于 16

D. 前三种情况都有可能

答案　A

解析

不难发现该算法在利用二分法求平方根。最后的输出同时对平方根进行了验证。

事实上，该算法为牛顿迭代法，solve1求解近似整数平方根，solve2进行 k 次牛顿迭代，找出更精确的平方根。但注意，如果不能开平方，无论多少次迭代都找不到精确值，即ans * ans 一定不等于 n。

1）solve1 在 n 范围内二分，时间复杂度为 logn；solve2 做 k 次牛顿迭代，时间复杂度为 O(k)。因此综合时间复杂度为 O(logn + k)。

2）有整数平方根，直接二分就搜到了。

3）n 是不一定有准确平方根的。因此 ans * ans 不一定等于 n。

4）n 不超过 47000，所以没必要。

5）2 没有有理数平方根，因此直接带入 n 和 k 逼近即可。不要想当然地选 B。逼近次数只有 1，很难那么准确。

6）同上一题，模拟几次即可得到选项。

7）整数平方根二分就算出来了，跟迭代都没关系。

三、完善程序

第 19 - 23 题　组合题

（枚举因数）从小到大打印正整数 n 的所有正因数。

试补全枚举程序。

```
01 #include <bits/stdc++.h>
02 using namespace std;
03
04 int main() {
05 int n;
06 cin >> n;
07
08 vector<int> fac;
09 fac.reserve((int)ceil(sqrt(n)));
10
11 int i;
```

青少年考级竞赛题库--在线刷题/真题模拟/不限科目/顺利通过-https://tiku.scratchor.com/

```
12 for (i = 1; i * i < n; ++i) {
13 if (①) {
14 fac.push_back(i);
15 }
16 }
17
18 for (int k = 0; k < fac.size(); ++k) {
19 cout << ② << " ";
20 }
21 if (③) {
22 cout << ④ << " ";
23 }
24 for (int k = fac.size() - 1; k >= 0; --k) {
25 cout << ⑤ << " ";
26 }
27 }
```

第 19 题　单选题

①处应填（ ）

A.　n % i == 0

B.　n % i == 1

C.　n % (i-1)== 0

D.　n % (i-1) == 1

答案　A

第 20 题　单选题

②处应填（ ）

A.　n / fac[k]

B.　fac[k]

C.　fac[k]-1

D.　n / (fac[k]-1)

答案　B

第 21 题　单选题

③处应填（ ）

A.　(i-1) * (i-1) == n

B.　(i-1) * i == n

C.　i * i == n

D.　i * (i-1) == n

答案　C

青少年考级竞赛题库--在线刷题/真题模拟/不限科目/顺利通过-https://tiku.scratchor.com/

第 22 题　单选题

④处应填（ ）

A.　n-i

B.　n-i+1

C.　i-1

D.　i

答案　D

第 23 题　单选题

⑤处应填（ ）

A.　n / fac[k]

B.　fac[k]

C.　fac[k]-1

D.　n / (fac[k]-1)

答案　A

解析

因数分解算法，NOI 算法 L4 课程内容。核心要点在于遍历到平方根，找到较小的一部分因数，而后利用除法找到较大的一部分因数。

1）n 能整除 i，说明 i 是 n 的因数。

2）fac里面存了较小的因数，输出出来。

3）这里是判断是否是完全平方数。完全平方数的平方根是重复的因数，因此要单独处理。注意到最后有 ++i，因此直接看 i 是不是平方根即可。

4）输出这个平方根 i。

5）利用除法计算出所有较大的因数并输出。

第 20 - 24 题　组合题

（洪水填充）现有用字符标记像素颜色的 8x8 图像。颜色填充的操作描述如下：给定起始像素的位置和待填充的颜色，将起始像素和所有可达的像素（可达的定义：经过一次或多次的向上、下、左、右四个方向移动所能到达且终点和路径上所有像素的颜色都与起始像素颜色相同），替换为给定的颜色。

试补全程序。

```
01 #include <bits/stdc++.h>
02 using namespace std;
03
04 const int ROWS = 8;
05 const int COLS = 8;
06
07 struct Point {
08     int r, c;
```

青少年考级竞赛题库--在线刷题/真题模拟/不限科目/顺利通过-https://tiku.scratchor.com/

```
09      Point(int r, int c) : r(r), c(c) {}
10 };
11
12 bool is_valid(char image[ROWS][COLS], Point pt,
13      int prev_color, int new_color) {
14      int r = pt.r;
15      int c = pt.c;
16      return (0 <= r && r < ROWS && 0 <= c && c < COLS &&
17      ① && image[r][c] != new_color);
18 }
19
20 void flood_fill(char image[ROWS][COLS], Point cur, int new_color) {
21      queue<Point> queue;
22      queue.push(cur);
23
24      int prev_color = image[cur.r][cur.c];
25      ② ;
26
27      while (!queue.empty()) {
28          Point pt = queue.front();
29          queue.pop();
30
31          Point points[4] = { ③ , Point(pt.r - 1, pt.c),
32          Point(pt.r, pt.c + 1), Point(pt.r, pt.c - 1)};
33          for (auto p : points) {
34              if (is_valid(image, p, prev_color, new_color)) {
35                  ④ ;
36                  ⑤ ;
37              }
38          }
39      }
40 }
41
42 int main() {
43 char image[ROWS][COLS] = {{'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g'},
44 {'g', 'g', 'g', 'g', 'g', 'g', 'r', 'r'},
45 {'g', 'r', 'r', 'g', 'g', 'r', 'g', 'g'},
46 {'g', 'b', 'b', 'b', 'b', 'r', 'g', 'r'},
47 {'g', 'g', 'g', 'b', 'b', 'r', 'g', 'r'},
48 {'g', 'g', 'g', 'b', 'b', 'b', 'b', 'r'},
49 {'g', 'g', 'g', 'g', 'g', 'b', 'g', 'g'},
50 {'g', 'g', 'g', 'g', 'g', 'b', 'b', 'g'}};
51
```

青少年考级竞赛题库--在线刷题/真题模拟/不限科目/顺利通过-https://tiku.scratchor.com/

```
52 Point cur(4, 4);
53 char new_color = 'y';
54
55 flood_fill(image, cur, new_color);
56
57 for (int r = 0; r < ROWS; r++) {
58 for (int c = 0; c < COLS; c++) {
59 cout << image[r][c] << " ";
60 }
61 cout << endl;
62 }
63 // 输出:
64 // g g g g g g g g
65 // g g g g g g r r
66 // g r r g g r g g
67 // g y y y y r g r
68 // g g g y y r g r
69 // g g g y y y y r
70 // g g g g g y g g
71 // g g g g g y y g
72
73 return 0;
74 }
```

第 20 题　单选题

①处应填（ ）

A.　image[r][c] == prev_color

B.　image[r][c] != prev_color

C.　image[r][c] == new_color

D.　image[r][c] != new_color

答案　A

第 21 题　单选题

②处应填（ ）

A.　image[cur.r+1][cur.c] = new_color

B.　image[cur.r][cur.c] = new_color

C.　image[cur.r][cur.c+1] = new_color

D.　image[cur.r][cur.c] = prev_color

答案　B

第 22 题　单选题

③处应填（ ）

A. Point(pt.r, pt.c)

B. Point(pt.r, pt.c+1)

C. Point(pt.r+1, pt.c)

D. Point(pt.r+1, pt.c+1)

答案 C

第 23 题 单选题

④处应填（ ）

A. prev_color = image[p.r][p.c]

B. new_color = image[p.r][p.c]

C. image[p.r][p.c] = prev_color

D. image[p.r][p.c] = new_color

答案 D

第 24 题 单选题

⑤处应填（ ）

A. queue.push(p)

B. queue.push(pt)

C. queue.push(cur)

D. queue.push(Point(ROWS,COLS))

答案 A

解析

经典的 BFS 洪水填充，甚至给出了起始坐标，没什么难点。

1）prev_color 是之前的颜色，new_color 是新的颜色，洪水填充要将之前的颜色改为新的颜色，因此要求 image[r][c] == prev_color。

2）起点入队之后，把起点颜色改成新的颜色。

3）经典 dx[] = {1, -1, 0, 0}, dy = {0, 0, 1, -1}。因此把 r + 1，c + 0补上即可。

4）找到了可以填充的节点，因此把颜色修改掉。

5）新节点入队即可。

青少年考级竞赛题库--在线刷题/真题模拟/不限科目/顺利通过-https://tiku.scratchor.com/