CCF 编程能力等级认证
Grade Examination of Software Programming

C++ 二级

2026 年 03 月

1 单选题（每题 2 分，共 30 分）

第 1 题 2026年春节联欢晚会上一个武术表演节目《武 BOT》。节目中多个人形机器人会表演空翻，它们落地可能会有微微踉跄，但都会迅速调整姿态站稳，并适当移动来和前后左右的其他机器人保持原来队列。如果将机器人视作一个计算机系统，那么在该计算机系统中下面哪一项不能作为输入设备( )。

- A. 检测重心的重力传感器
- B. 预装的AI算法程序
- C. 接收动作指令的遥控器
- D. 拍摄其他机器人的摄像头

第 2 题 下面代码用来找出输入的N个正整数中最大的一个。如果将代码段用流程图来表示，则 `L1` 标记的代码行应该使用的图形是（ ）。

```
int N, max=0, val;
cin >> N;

while(N){
    cin >> val;
    if(val > max) // L1
        max = val;
    N--;
}

cout << max;
```

- A. 圆形框
- B. 椭圆形框
- C. 平行四边形框
- D. 菱形框

第 3 题 下面 C++ 代码可以执行，有关说法正确的是( )。

```
double PI = 3.1415926;
cout << (PI);
```

- A. 为了方便初学者， `cout << (PI)` 和 `cout << (pi)` 效果相同，即变量的大小写不敏感
- B. `cout << (PI)` 修改为 `cout << (Pi)` 能正常执行

- [ ] **C.** 不能用 `PI` 做变量名，因为要保存圆周率这个常量
- [ ] **D.** 将程序中全部 `PI` 都改写为 `Pai` ，将能正常执行，不会报错

第 4 题 下面选择项中，与C++表达式 `!(x > 5 && y <= 10)` 等价的是( )。

- [ ] **A.** `x <= 5 && y > 10`
- [ ] **B.** `x > 5 || y <= 10`
- [ ] **C.** `x <= 5 || y > 10`
- [ ] **D.** `! x > 5 && ! y <= 10`

第 5 题 某同学执行C++代码 `cout << ((0.1 + 0.2) == 0.3)` 时输出 `0` ，下面最可能的原因是( )。

- [ ] **A.** C++的 `+` 运算符在处理小数时存在bug
- [ ] **B.** `0.1` 、 `0.2` 和 `0.3` 在计算机中无法用二进制浮点数精确表示，导致 `0.1 + 0.2` 的结果与 `0.3` 存在微小误差
- [ ] **C.** `==` 运算符不能用于比较浮点数，只能用于整数
- [ ] **D.** 因为 `0.1 + 0.2` 的数学结果不等于 `0.3`

第 6 题 下面的 C++ 代码段执行后其输出是( )。

```
tnt = 0;
for (int i = 0; i < 5; i++){
    for (int j = 0; j < i; j++)
        tnt += 1;
    cout << tnt << "#";
}
cout << tnt;
```

- [ ] **A.** `0#1#3#6#10#10`
- [ ] **B.** `1#2#3#4#5#6#7#8#9#10#10`
- [ ] **C.** `10#10`
- [ ] **D.** `10`

第 7 题 下面的 C++ 代码执行之后的输出是( )。

```
for ( int i = -2; i < 2; i++)
    if (not i % 3)
        cout << i << "#";
```

- [ ] **A.** `0#`
- [ ] **B.** `-2#-1#1#`
- [ ] **C.** `-1#0#`
- [ ] **D.** `-2#0#1#`

第 8 题 下面的 C++ 代码执行后其输出是( )。

```
int cnt = 0, i, j;
for (i =1; i < 5; i++){
    for (j =0; j < i; j++)
       cout << j << "#";
    break;
}
if(i >= 5)
    cout << (i*j);
```

- [ ] **A.** `0#0#1#0#1#2#0#1#2#3#12`
- [ ] **B.** `0#0#1#0#1#2#0#1#2#3#`
- [ ] **C.** `0#`
- [ ] **D.** `1#`

第 9 题 下面 C++ 代码执行后其输出是(　)。

```
int count = 0;
for (int i = 1; i < 4; i++)
    for (int j = 1; j < 5; j++){
        if (j == 3)
            continue;
        if (i == 2)
            break;
        count += 1;
    }
cout << (count);
```

- [ ] **A.** 2
- [ ] **B.** 4
- [ ] **C.** 6
- [ ] **D.** 8

第 10 题 下面4个选项中，与下面 C++ 代码段具有相同效果的是(　　)。

```
i = 0;
while (i < 5){
    cout << i;
    i += 1;
}
```

- [ ] **A.**

```
for (i = 0; i < 5; i++)
    cout << i;
```

- [ ] **B.**

```
for (i = 1; i < 5; i++)
    cout << i;
```

- [ ] **C.**

```
for (i = 0; i < 6; i++)
    cout << i;
```

D.

```
for (i = 1; i < 6; i++)
    cout << i;
```

第 11 题 下面 C++ 代码执行后输出是（　）。

```
int n = 10;
while (n > 0){
    n -= 1;
    if (n % 3 == 0)
        continue;
    if (n == 5)
        break;
}
cout << n;
```

A. 0

B. 5

C. 6

D. 7

第 12 题 下面 C++ 代码段执行后，其输出是（　）。

```
int i, j, cnt;
cnt = 0;
for (i = 0; i < 5; i++){
    i = -i;
    for (j = i; j < -i; j++)
        cnt += 1;
    i = -i;
}
cout << cnt;
```

A. 5

B. 15

C. 20

D. 30

第 13 题 某学校图书馆的借阅卡号由6位整数组成。前5位是顺序编号，第6位是校验码，用于防止输错。校验码规则如下：将前5位数字相加，然后除以10的余数，就是第6位数字。如卡号 `123455` 的前5位之和为 `15`，除以 `10` 的余数是5，故第6位为5。下面的C++代码段用于判断卡号是否正确，横线处应填入的代码是(　　)。

```
cout << "请输入卡号：";
cin >> N;
order_num = N / 10; // 获得前5位顺序号,假设录入一定为6位正整数
check_num = N % 10; // 获得最后一位

tnt = 0; // 保存前5位之和
for (i = 0; i < 5; i++){
    _________________;
    order_num /= 10;
}

if (___________________)
    cout << "符合校验规则";
else
    cout << "不符合校验规则";
```

☐ **A.**

```
tnt += order_num / 10
tnt / 10 == check_num
```

☐ **B.**

```
tnt += order_num % 10
tnt % 10 == check_num
```

☐ **C.**

```
tnt = order_num / 10 + tnt
tnt % 10 == check_num
```

☐ **D.**

```
tnt = order_num % 10
tnt / 10 == check_num
```

第 14 题 下面的 C++ 代码段正常执行后其输出的数字图形是（ ）。

```
for (i = 1; i < 5; i++){
    for (j = 1; j < i+1; j++)
        cout << j;
    cout << endl;
}
```

☐ **A.**

```
1
12
123
1234
```

☐ **B.**

```
1
22
333
4444
```

C.

```
1
21
321
4321
```

D.

```
4
34
234
1234
```

第 15 题 某学校举办“校园演讲比赛”，每位选手由8位评委打分（分数为 0~10 的整数），且每位评委必须打分。计分规则：若至少有5位评委给出大于等于6分，则成绩有效，最终得分为所有8位评委的总分；如给出低于6分的评委数量超过5位，则记为0分。以下核心程序段依次输入8个分数，并计算最终得分。横线处应填入（ ）。

```
total_score = 0; // 所有分数之和
high_count = 0;  // ≥6分的评委数量

for (i = 0; i < 8; i++){
    cout << "请输入评委分数: ";
    cin >> score;
    ____________________;
    if (score >= 6)
        ________________;
}

if (high_count >= 5)
    cout << total_score;
else
    cout << 0;
```

A.

```
total_score += score
high_count += 1
```

B.

```
total_score += score
high_count += score
```

C.

```
high_count += 1
total_score += score
```

D.

```
total_score *= score
high_count *= 1
```

2 判断题（每题 2 分，共 20 分）

第 1 题 小明的妈妈最近刚刚给他买了一块电话手表，除了可以看时间，小明也可以用它和妈妈打电话、收发信息，那么可以推测这块手表中装有一款特定操作系统。（ ）

第 2 题 C++ 语句 `cout << ('4' % '2' == '2' * '2' % 2);` 执行后的输出是1。(　)

第 3 题 下面的 C++ 代码段执行时将报错，因为10是整数类型，`a` 是布尔类型。由于类型不同，不能进行加法运算。（ ）

```
bool a = true;
cout << (10 + a);
```

第 4 题 下面C++代码段执行后将输出 `0-3-6-9-` 。(　)

```
for (int i = 0; i < 10; i++){
    if (i % 3)
        continue;
    cout << i <<  "-";
}
```

第 5 题 执行下面的 C++ 代码段，如果N是基本数据类型，则语句 `cout << (N);` 将被执行 0 次或无数次（即死循环）。（ ）

```
cin >> N;
while (N)
    cout << (N);
```

第 6 题 下面的 C++ 代码段可正常执行，删除 `continue` 不影响执行效果。（ ）

```
for (i = 0; i < 10; i++){
    i += 1;
    continue;
}
cout << (i);
```

第 7 题 下面的 C++ 代码段用于计算1到N之间且包含N的所有数字中含有的3的个数，如1到40含有14个3, 而1到20则只含有2个3。如果将 `while (i != 0)` 改为 `while (abs(i))` ，则执行结果相同。(　)

```
cout << "请输入正整数N: ";
cin >> N;
cnt = 0; // 保存3的个数
for  (k = 1; k < N + 1; k++){
    i = k;
    while (i != 0){
        if (i % 10 == 3)
            cnt += 1;
        i /= 10;
     }
}
cout << cnt;
```

第 8 题 下面的 C++ 代码段执行将不会有输出，因为内层循环 `j` 总是 0 开始，`i * j % 10 == 0` 将会被满足，执行 `break` ，故而 `i` 小于 `10` ，不会满足 `if` 判断条件。（ ）

```
for (i = 1; i < 10; i++)
    for (j = 0; j < i; j++)
        if (i * j % 10 == 0)
            break;
if(i >= 10)
    cout << (i*j);
```

第 9 题 下列 C++ 代码执行后将输出 `1#4#9#16#16` 。（ ）

```
cnt = 0;
for  (i = 1; i < 5; i++){
    for (j = 1; j < i + 1; j++)
        if (i * j % 10 == 0)
            break;
    if (j >= i + 1)
        cout << (i * j) << "#";
}
if(i >= 5)
    cout << (i * j);
```

第 10 题 下面 C++ 代码执行后输出如左图所示，将 `" %d"` 修改为 `"%3d"` 即可实现右图输出。（ ）

```
/*
  // 左图                          // 右图
 1 2 3 4 5 6 7 8 9                1   2   3   4   5   6   7   8   9
 2 4 6 8 10 12 14 16 18           2   4   6   8  10  12  14  16  18
 3 6 9 12 15 18 21 24 27          3   6   9  12  15  18  21  24  27
 4 8 12 16 20 24 28 32 36         4   8  12  16  20  24  28  32  36
 5 10 15 20 25 30 35 40 45        5  10  15  20  25  30  35  40  45
 6 12 18 24 30 36 42 48 54        6  12  18  24  30  36  42  48  54
 7 14 21 28 35 42 49 56 63        7  14  21  28  35  42  49  56  63
 8 16 24 32 40 48 56 64 72        8  16  24  32  40  48  56  64  72
 9 18 27 36 45 54 63 72 81        9  18  27  36  45  54  63  72  81
*/

for (i = 1; i < 10; i++){
    for (j = 1; j < 10; j++)
        printf(" %d", i*j);
    printf("\n");
}
```

3 编程题（每题 25 分，共 50 分）

3.1 编程题 1

- **试题名称**：数数
- **时间限制**：1.0 s
- **内存限制**：512.0 MB

3.1.1 题目描述

对于正整数 n，如果 n 的所有数位中包含**恰好** 3 个 2，Alice 会认为这个正整数是美丽的。例如，正整数 24122 中包含 3 个 2，所以 24122 **是**美丽的；正整数 132 中包含 1 个 2，所以 132 **不是**美丽的；正整数 212322 中包含 4 个 2，所以 212322 **不是**美丽的。

Alice 想知道正整数 L 到正整数 R （包括 L 和 R）中有多少美丽的正整数，你能帮她数一数吗?

3.1.2 输入格式

输入共 2 行，第一行为正整数 L，第二行为正整数 R。

3.1.3 输出格式

输出一行，包含一个整数 C，表示 L 到 R 中 C 美丽数的数量。

3.1.4 样例

3.1.4.1 输入样例

```
2221
2223
```

3.1.4.2 输出样例

```
2
```

3.1.5 样例解释

2221 到 2223 中，2221 与 2223 是美丽的，2222 不是美丽的。

3.1.6 数据范围

保证 $1 \leq L \leq R \leq 10^6$。

3.2 编程题 2

- **试题名称**：画画
- **时间限制**：1.0 s
- **内存限制**：512.0 MB

3.2.1 题目描述

输入一个正整数 n，你需要绘制一个 n 行 n 列的正方形，绘制规则如下：

- 正方形的四个顶点使用 `+` 绘制；
- 除顶点外，第 1 行与第 n 行使用 `-` 绘制；
- 除顶点外，第 1 列与第 n 列使用 `|` 绘制；
- 正方形内部使用 `*` 绘制。

3.2.2 输入格式

一行，一个正整数 n。

3.2.3 输出格式

输出共 n 行，表示对应的正方形。

3.2.4 样例

3.2.4.1 输入样例

```
5
```

3.2.4.2 输出样例

```
+---+
|***|
|***|
|***|
+---+
```

3.2.5 数据范围

保证 $3 \leq n \leq 100$。