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

C++ 二级

2026 年 06 月

1 单选题（每题 2 分，共 30 分）

第 1 题 学校组织到未来农场参观，小明听讲解员介绍，在智能温室中，湿度传感器可以连续检测土壤的湿度变化，并将检测到的湿度数据实时发送给中央控制器。中央控制器根据这些数据判断是否开启灌溉系统。请问，这里的湿度传感器所发挥的作用，类似于计算机系统中的哪一类组件？（ ）

- A. 处理器
- B. 存储器
- C. 输入设备
- D. 输出设备

第 2 题 小明去农场参观回来后就在自己的电脑上安装了一个 3D 农场仿真模拟系统，因为他今年买的二手电脑有点老旧，系统运行时经常弹出“系统内存不足”的警告，导致系统卡顿严重。他计划通过增加物理内存来解决问题。请问他应该购买以下哪种硬件组件？（ ）

- A. 机械硬盘
- B. 内存条
- C. 图形显卡
- D. 移动硬盘

第 3 题 有关如下 C++ 代码的说法，正确的是（ ）。

```
1 float a = 3 + 3.5;
2 cout << a;
```

- A. 代码执行将报错。如果将 `a = 3 + 3.5` 改为 `a = float(3) + 3.5` 将能正常执行。
- B. 代码执行将报错。如果将 `a = 3 + 3.5` 改为 `a = 3.0 + 3.5` 将能正常执行。
- C. 代码能正常执行，将输出 `6`。
- D. 代码能正常执行，将输出 `6.5`。

第 4 题 下面选择项中，与 C++ 表达式 `not (x > 5 or y <= 10)` 等价的是（ ）。

- A. `x <= 5 or y > 10`
- B. `x > 5 and y <= 10`
- C. `x <= 5 and y > 10`
- D. `not x > 5 and not y <= 10`

第 5 题 小明在某开发环境中执行 C++ 代码 `cout << (2.5 + 2.25) << ' ' << (2.2 + 2.1)` 时输出 `4.75 4.300000000000001`，其原因最可能是（ ）。

- [ ] **A.** C++ 的 `+` 运算符在处理浮点数时有时正确，有时错误
- [ ] **B.** 某些浮点数难以精确表示，导致微小误差
- [ ] **C.** `+` 运算符不能用于浮点数，只能用于整数
- [ ] **D.** 因为 `cout` 函数难以输出太长的数值

第 6 题 执行如下 C++ 程序后，当输入 `4` 时，输出的最后一行是（ ）。

```cpp
int n;
cin >> n;
for (int i = n; i > 0; i--) {
    for (int j = 0; j < i; j++)
        cout << j + 1 << ' ';
    cout << endl;
}
```

- [ ] **A.** `0`
- [ ] **B.** `1`
- [ ] **C.** `1 2`
- [ ] **D.** `1 2 3 4`

第 7 题 下面的 C++ 代码执行后其输出是（ ）。

```cpp
int tnt = 0;
for (int i = 1; i < 5; i += 3) {
    for (int j = 0; j < i; j++)
        tnt += 1;
    cout << tnt << "#";
}
cout << tnt;
```

- [ ] **A.** `1#5#5`
- [ ] **B.** `1#5#5#`
- [ ] **C.** `1#5#12#12`
- [ ] **D.** `0`

第 8 题 下面的 C++ 代码执行之后的输出是（ ）。

```cpp
int i;
for (i = -2; i < 2; i++)
    if (not i % 3 == 0)
        cout << i << "#";
cout << i;
```

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

第 9 题 下面的 C++ 代码执行后其输出是（ ）。

```cpp
int cnt = 0, i, j;
for (i = 0; i < 3; i++) {
    j = i;
    while (j) {
        cnt += 1;
        j -= 1;
    }
}
cout << i << ' ' << j << ' ' << cnt;
```

- [ ] **A.** `3 0 3`
- [ ] **B.** `2 0 4`
- [ ] **C.** `2 0 5`
- [ ] **D.** `3 0 5`

第 10 题 下面 C++ 代码执行后其输出是（ ）。

```cpp
int count = 0, i, j;
for (i = 1; i < 4; i++)
    for (j = 0; j < i; j++) {
        if (j % 3 != 0)
            continue;
        count += 1;
        break;
    }
cout << i << ' ' << j << ' ' << count;
```

- [ ] **A.** `4 0 4`
- [ ] **B.** `3 0 0`
- [ ] **C.** `3 2 0`
- [ ] **D.** `4 0 3`

第 11 题 如下 C++ 代码执行后输出是 `1 2 3 4 5 6 7 8 9 10 11 5 6 7 8 9 10 11 5 6 7 8 9 10 11 5 6 7 8 9 10 11 5 6 7`。横线处应该填入的运算符是（ ）。

```cpp
int num = 1;
for (int i = 0; i < 35; i++) {
    cout << num << " ";
    if (num _______ 10)
        num _______ 2;
    else
        num _______ 1;
}
```

- [ ] **A.** `>` `/=` `+=`
- [ ] **B.** `>=` `%=` `+=`
- [ ] **C.** `>` `/=` `=+`
- [ ] **D.** `>=` `%=` `=+`

第 12 题 如下数字图形在通过执行后续的 C++ 代码时，输入 `10` 来输出。横线处应填入的代码是（ ）。

```
1 1 1 1 1 1 1 1 1 1
1 1 0 0 0 0 0 0 0 1
1 0 1 0 0 0 0 0 0 1
1 0 0 1 0 0 0 0 0 1
1 0 0 0 1 0 0 0 0 1
1 0 0 0 0 1 0 0 0 1
1 0 0 0 0 0 1 0 0 1
1 0 0 0 0 0 0 1 0 1
1 0 0 0 0 0 0 0 1 1
1 1 1 1 1 1 1 1 1 1
```

```cpp
int N;
cin >> N;
for (int i = 1; i < N + 1; i++) {
    for (int j = 1; j < N + 1; j++)
        if (________)
            cout << "1 ";
        else
            cout << "0 ";
    cout << endl;
}
```

- [ ] **A.** `i == j and i == 1 and j == 1 and i == N and j == N`
- [ ] **B.** `i == j or i == 1 or j == 1 or i == N or j == N`
- [ ] **C.** `i == j or i == 0 or j == 0 or i == (N + 1) or j == (N + 1)`
- [ ] **D.** `i == j and i == 1 and j == 1 and i == (N + 1) and j == (N + 1)`

第 13 题 英文 `corner case` 通常翻译为极端案例或边角案例，通常指正常范围以外的问题或者情形。在如下 C++ 代码中，变量都是整型，则 `corner case` 最应该是（ ）。

```cpp
int tnt = 0, cnt = 0;
while (1) {
    int score;
    cin >> score;
    if (score == -1)
        break;
    tnt += score;
    cnt += 1;
}
cout << tnt / cnt;
```

- [ ] **A.** `tnt = 0, cnt = 0` 是 `corner case`，应分为两行
- [ ] **B.** `while(1)` 是 `corner case`，因为 `while(1)` 将会导致死循环
- [ ] **C.** `cin >> score` 是 `corner case`，因为输入数据前应该提示整型数据
- [ ] **D.** `cout << tnt / cnt` 是 `corner case`，因为如果直接输入 `-1`，将导致执行错误，虽然这种情况较为罕见

第 14 题 如下 C++ 代码执行后，输入 `4` 后，输出的数字图形是（ ）。

```
int n;
cin >> n;
for (int i = n; i > 0; i--) {
    for (int j = 0; j < n - i; j++)
        cout << "0 ";
    for (int k = 0; k < i; k++)
        cout << k + 1 << " ";
    cout << endl;
}
```

- [ ] **A.**

```
1 2 3 4
1 2 3 0
1 2 0 0
1 0 0 0
```

- [ ] **B.**

```
1 2 3 4
0 1 2 3
0 0 1 2
0 0 0 1
```

- [ ] **C.**

```
1 2 3 4
2 3 4 0
3 4 0 0
4 0 0 0
```

- [ ] **D.**

```
0 0 0 1
0 0 1 2
0 1 2 3
1 2 3 4
```

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：去掉一个最高分，去掉一个最低分。如下程序通过键盘先输入选手编号，然后依次输入 8 个分数，并计算最终得分。下列说法正确的是（ ）。

```c
for (int i = 0; i < 10; i++) {
    int id, score;
    printf("输入选手编号: ");
    scanf("%d", &id);

    int max_score = 0, min_score = 100; // 最高分和最低分
    int total_score = 0; // 总分

    for (int j = 1; j < 9; j++) {
        printf("输入选手第%d个成绩:", j);
        scanf("%d", &score);

        if (max_score < score)
            max_score = score;

        if (min_score > score)
            min_score = score;

        total_score += score;
    }

    total_score = total_score - max_score - min_score;
    printf("%d号选手的成绩:\
            去掉一个最高分%d,\
            去掉一个最低分%d,\
            最后成绩是: %d", id, max_score, min_score, total_score);
}
```

- ☐ **A.** 上述代码能完成题目要求
- ☐ **B.** `max_score = 0, min_score = 100` 应修改为 `max_score = 0, min_score = 0`
- ☐ **C.** `max_score < score` 和 `min_score > score` 必须相应修改为 `<=` 和 `>=`
- ☐ **D.** `total_score = total_score - max_score - min_score` 不能达到预期，可修改如下：

```c
int total = total_score - max_score - min_score;
total_score = total;
```

2 判断题（每题 2 分，共 20 分）

第 1 题 又到期末考试周，小明发现这次许多闭卷考试不仅禁止携带手机、平板电脑，还有最近比较时髦的各类 AI 眼镜（也有叫智能眼镜）也同样不允许带入考场。这些 AI 眼镜应该也是内置了操作系统并可能支持 Wi-Fi 或蓝牙连接。

第 2 题 C++ 代码 `cout << (not ('5' % 2 == 0) == ((not '5' % 2) == 0))` 执行后的输出是 `1`。

第 3 题 执行 C++ 语句 `cout << (int(3.5) * 2)` 将输出 `6`。

第 4 题 下面 C++ 代码执行后将输出 `1-4-7-`。

```c
for (int i = 1; i < 10; i += 3) {
    if (not i % 3)
        break;
    cout << i << "-";
}
```

第 5 题 执行如下 C++ 代码，将从小到大依次输出 `abs(N)` 个整数，并在最后输出 `1` 。

```cpp
int N;
cin >> N;
int start_num = 1, end_num = N + 1, i;

if (N < 0)
    start_num = N, end_num = 0;

for (i = start_num; i < end_num; i++)
    cout << i << " ";

cout << endl << ((i - 1) == abs(N)) << endl;
```

第 6 题 如下 C++ 代码执行后，输出值为 `9` 。

```cpp
int cnt = 0, i;
for (i = 0; i < 10; i++) {
    for (int j = 0; j < i; j++)
        cnt += 1;
    break;
}
cout << i;
```

第 7 题 如下 C++ 代码执行时如输入 `10` ，输出将是 `100` 。

```cpp
int cnt = 0, N;
cout << "请输入正整数: ";
cin >> N;
for (int i = 0; i < N; i++)
    for (int j = -i; j < i; j++)
        cnt += 1;
cout << cnt;
```

第 8 题 如下 C++ 代码执行其输出是 `3` 。

```cpp
int count = 0;
int i = 0;
while (i < 3) {
    int j = 0;
    while (j < 3) {
        if (i + j >= 3)
            count += 1;
        j += 1;
    }
    i += 1;
}
cout << count;
```

第 9 题 如下 C++ 代码执行时如果输入正整数，其输出将是输入的正整数。

```cpp
int N;
cin >> N;
int i = 0, Nbase = 1;
int rst = 0;
while (N != 0) {
    rst = rst + N % 10 * Nbase;
    N /= 10, Nbase *= 10;
    i += 1;
}
cout << rst;
```

第 10 题 如下 C++ 代码执行时如输入 `5`，将输出代码后的字符图形。

```cpp
int n;
cin >> n;
for (int i = 1; i < n + 1; i++) {
    for (int j = 1; j < n - i + 1; j++)
        cout << 0;
    for (int k = 1; k < 2 * i; k++) {
        if (k <= i)
            cout << k;
        else
            cout << 2 * i - k;
    }
    for (int j = 1; j < n - i + 1; j++)
        cout << 0;
    cout << endl;
}
```

```
000010000
000121000
001232100
012343210
123454321
```

3 编程题（每题 25 分，共 50 分）

3.1 编程题 1

- **试题名称**：完全平方数计数
- **时间限制**：1.0 s
- **内存限制**：512.0 MB

3.1.1 题目描述

小杨同学正在研究**完全平方数**。

平方：一个数的平方等于这个数乘以这个数本身。

完全平方数：指可以恰好表示为某个正整数的平方的数。

例如，9 是完全平方数，因为 $9 = 3^2 = 3 \times 3$；但 27 不是，因为 27 不能表示为任何正整数的平方。

给定两个正整数 l 和 r（保证 $l \leq r$），小杨同学想知道 l 到 r 之间的所有正整数中（包含 l 和 r），有多少个数是**完全平方数**。

3.1.2 输入格式

输入两行，第一行为一个正整数 l，第二行为一个正整数 r。

3.1.3 输出格式

输出一个非负整数，表示 l 到 r 中，有多少个正整数是**完全平方数**。如果 l 到 r 中没有完全平方数，则输出 `0` 。

3.1.4 样例

3.1.5 输入样例 1

```
1
21
```

3.1.6 输出样例 1

```
4
```

3.1.7 样例解释 1

在 1 到 21 中，有以下 4 个整数是**完全平方数**：
$1, 4, 9, 16$

3.1.8 数据范围

$1 \leq l \leq r \leq 2000$。

3.2 编程题 2

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

3.2.1 题目描述

给定正整数 n，在 $(2n-1) \times (2n-1)$ 个网格的画布中，使用字符画一个边长为 n 个网格的菱形。其中，空白网格使用 `.` 表示，菱形边所在的网格用 `+` 表示。

例如当 $n = 3$ 时，图形如下：

```
..+..
.+.+.
+...+
.+.+.
..+..
```

3.2.2 输入格式

输入一个正整数 n;

3.2.3 输出格式

输出 $2n-1$ 行，表示按要求画的菱形。

3.2.4 样例

3.2.5 输入样例 1

```
4
```

3.2.6 输出样例 1

```
...+...
..+.+..
.+...+.
+.....+
.+...+.
..+.+..
...+...
```

3.2.7 数据范围

$3 \leq n \leq 15$。