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

Python 二级

2026 年 06 月

1 单选题（每题 2 分，共 30 分）

题号	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
答案	C	B	D	C	B	B	A	A	A	D	A	B	D	B	A

第 1 题 学校组织同学们到未来农场参观，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，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 小明同学去郊外农场参观回来后，在自己的电脑中安装了一个 3D 农场仿真模拟系统。因为他今年买的这台二手电脑有点老旧，3D 农场仿真模拟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 Python 代码的说法，正确的是（ ）。

```
a = 3 + 3.5
print(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 Python 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 or not y <= 10`

第 5 题 某同学执行 Python 代码 `print(2.5+2.25,2.2+2.1)` 时输出 `4.75 4.300000000000001`，其原因是（ ）。

- [ ] **A.** Python 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 `print()` 函数难以输出太长的数值

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

```python
n = int(input())
for i in range(n, 0, -1):
    for j in range(i):
        print(j + 1, end='#')
    print()
```

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

第 7 题 下面的 Python 代码执行后其输出是（ ）。

```python
tnt = 0
for i in range(1, 5, 3):
    for j in range(i):
        tnt += 1
    print(tnt, end = "#")
print(tnt)
```

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

第 8 题 下面的 Python 代码执行后的输出是（ ）。

```python
for i in range(-2, 2):
    if not i % 3 == 0:
        print(i, end = "#")
print(i)
```

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

第 9 题 下面的 Python 代码执行后其输出是（ ）。

```
cnt = 0
for i in range(3):
    j = i
    while j:
        cnt += 1
        j -= 1
print(i, j, cnt)
```

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

第 10 题 下面 Python 代码执行后其输出是（ ）。

```
count = 0
for i in range(1, 4):
    for j in range(i):
        if j % 3 != 0:
            continue
        count += 1
        break
print(i, j, count)
```

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

第 11 题 如下 Python 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` 之值。横线处应该填入的运算符是（ ）。

```
num = 1
for i in range(35):
    print(num, end = " ")
    if num ______ 10:
        num _______ 2
    else:
        num _______ 1
```

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

第 12 题 如下的 Python 代码用于输出如下数字图形。横线处应填入的代码是（ ）。

```
10
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
```

```python
N = int(input())
for i in range(1, N + 1):
    for j in range(1, N + 1):
        if ______________________________________:
            print(1, end = " ")
        else:
            print(0, end = " ")
    print()
```

- [ ] **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 `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是情形。在如下 Python 代码中，`corner case` 是（ ）。

```python
tnt, cnt = 0, 0
while True:
    score = int(input())
    if score == -1:
        break
    tnt += score
    cnt += 1
print(tnt / cnt)
```

- [ ] **A.** `tnt, cnt = 0, 0` 是 `corner case`，应分为两行
- [ ] **B.** `while True` 是 `corner case`，因为 `while True` 将会导致死循环
- [ ] **C.** `score = int(input())` 是 `corner case`，因为 `input()` 应该有提示信息
- [ ] **D.** `print(tnt / cnt)` 是 `corner case`，因为如果直接录入 `-1`，将导致错误，虽然这种情况较为罕见

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

```python
n = int(input())
for i in range(n, 0, -1):
    for j in range(n - i):
        print(0, end=" ")
    for k in range(i):
        print(k + 1, end=" ")
    print()
```

☐ **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（ ）。

```python
for _ in range(10):

    id = int(input("输入选手编号:"))

    max_score, min_score = 0, 100 # 最高分和最低分
    total_score = 0 # 总分

    for i in range(1, 9):
        score = int(input(f"输入选手第{i}个成绩:"))

        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
    print(f"""
            {id}号选手的成绩:
            去掉一个最高分{max_score},
            去掉一个最低分{min_score},
            最后成绩是:{total_score}""")
```

- [ ] **A.** 上述代码能完成题目要求
- [ ] **B.** `max_score, min_score = 0, 100` 应修改为 `max_score, min_score = 0, 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：

```python
total = total_score - max_score - min_score
total_score = total
```

2 判断题（每题 2 分，共 20 分）

题号	1	2	3	4	5	6	7	8	9	10
答案	√	√	×	√	√	×	×	√	√	√

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 Python 代码 `print(not (5 % 2 == 0) == ((not 5 % 2) == 0))` 执行后的输出是 `True`。

第 3 题 执行 Python 代码 `print(not 5)` 将 **报错**。

第 4 题 下面 Python 代码执行后将输出 `1-4-7-`。

```python
for i in range(1,10,3):
    if not i % 3:
        break
    print(i, end = "-")
```

第 5 题 执行如下 Python 代码，将从小到大依次输出共 $|N|$ 个整数，这里 $|N|$ 表示 N 的绝对值，如 $|-3|=3$。

```python
N = int(input())
start_num, end_num = 1, N + 1

if N < 0:
    start_num, end_num = N, 0

for i in range(start_num, end_num):
    print(i, end = " ")
```

第 6 题 如下 Python 代码执行后，输出值为 `9` 。

```python
cnt = 0
for i in range(10):
    for j in range(i):
        cnt += 1
    break
print(i)
```

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

```python
cnt = 0
N = int(input("请输入正整数: "))
for i in range(N):
    for j in range(-i, i):
        cnt += 1
print(cnt)
```

第 8 题 如下 Python 代码执行其输出是 `3` 。

```python
count = 0
i = 0
while i < 3:
    j = 0
    while j < 3:
        if i + j >= 3:
            count += 1
        j += 1
    i += 1
print(count)
```

第 9 题 执行如下 Python 代码时，如果输入正整数，其输出为该正整数。

```python
N = int(input())
i = 0
rst = 0
while N != 0:
    rst = rst + N % 10 * 10 ** i
    N //= 10
    i += 1
print(rst)
```

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

```python
n = int(input())
for i in range(1, n + 1):
    for j in range(1, n - i + 1):
        print(0, end='')
    for k in range(1, 2 * i):
        if k <= i:
            print(k, end='')
        else:
            print(2 * i - k, end='')
    for j in range(1, n - i + 1):
        print(0, end='')
    print()
```

```
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

$1 \le l \le r \le 2000$。

3.1.9 参考程序 1

```python
l = int(input())
r = int(input())
ans = 0
# 遍历每一个l和r之间的数
for n in range(l, r + 1):
    x = 1
    # 对于每一个数，我们看能不能找到一个x使得x**2==n
    while x * x < n:
        x += 1
    if x * x == n:
        ans += 1
print(ans)
```

3.1.10 参考程序 2

```python
# 输入左右边界
l = int(input())
r = int(input())
ans = 0 # 最终答案：有ans个完全平方数
# 因为r<2000，因此如果l<=i*i<=r的话，i只用考虑0~100即可
for i in range(100):
    if i * i > r: # 超过右边界r
        break
    # 大于等于左边界l，这样的i*i符合题意，给最终结果+1
    if i * i >= l:
        ans += 1
print(ans)
```

3.1.11 参考程序 3

```python
l = int(input())
r = int(input())
# 找出最小的整数a，a**2>=l
a = int(l**0.5)
if a**2 < l:
    a+=1
# 找出最大的整数b，b**2<=r
b = int(r**0.5)
print(max(0, b - a + 1))
```

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

$3\le n\le 15$。

3.2.8 参考程序

```python
n = int(input())
# 外层遍历，遍历行数
for i in range(1, 2 * n):
    # 内层遍历，遍历列数
    for j in range(1, 2 * n):
        # 是否处于菱形的左上角那条边
        if i + j == n + 1:
            print('+', end='')
        # 是否处于菱形的右下角那条边
        elif i + j == 3 * n - 1:
            print('+', end='')
        # 是否处于菱形的左下角或者右上角那条边
        elif i - j == n - 1 or j - i == n - 1:
            print('+', end='')
        # 其它的位置都是'.'
        else:
            print('.', end='')
    print()
```