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

Python 一级

2026 年 09 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	D	A	C	B	A	A	D	A	D	D	B	C	B	C

第 1 题 第二届世界人形机器人运动会开幕式上，机器人方阵通过控制程序来完成整场表演。表演前，工程师们使用集成开发环境（IDE）编写并调试方阵的控制程序。下列关于 IDE 功能的描述，错误的是（　　）。

- A. 可以提供代码自动补全，提高编程效率
- B. 可以提供辅助信息，帮助排查程序逻辑错误
- C. 可以直接控制机器人电机转动
- D. 支持语法高亮，使代码更易阅读

第 2 题 现代计算机自问世至今，已经经历了四个时代，即：电子管计算机（1946—1958 年）、晶体管计算机（1958—1964 年）、中小规模集成电路计算机（1965—1970 年）以及大规模和超大规模集成电路计算机（1971 年至今）。那么这里划分计算机时代所采用的主要依据是（　　）。

- A. 运算速度
- B. 存储容量
- C. 体积大小
- D. 核心元器件

第 3 题 下列 Python 代码执行后，其输出是（　　）。

```
a = 4
print(f"a+2={a+2}")
```

- A. `a+2=6`
- B. `a+2={a+2}`
- C. `6=6`
- D. `4+2=6`

第 4 题 下列 Python 表达式与 `2 * - 2 ** 4` 的值相同的是（　　）。

- A. `(2 * - 2) ** 4`
- B. `2 * (- 2) ** 4`
- C. `2 * - (2 ** 4)`
- D. `- (2 * 2) ** 4`

第 5 题 下列 Python 代码执行后，其输出是（　　）。

```
a = 3
b = 4
print(a, b, sep = "#")
```

- A. `3 4`
- B. `3#4`
- C. `3 # 4`
- D.

```
3
4
```

第 6 题 下列 Python 代码执行后，其输出是（　　）。

```
a = 8
b = 10
print(a % b, a / b, a // b )
```

- A. `8 0.8 0`
- B. `0 0.8 0`
- C. `8 0 0.8`
- D. `0 0 0.8`

第 7 题 下列 Python 代码中，`a * a + b * b == c * c` 可替换为（　　）。

```
a = int(input())
b = int(input())
c = int(input())

if a * a + b * b == c * c:
    print("1")
```

- A. `a ** 2 + b ** 2 == c ** 2`
- B. `a ** 2 + b ** 2 = c ** 2`
- C. `a ** a + b ** b = c ** 2`
- D. `a ** a + b ** b == c ** c`

第 8 题 下列 Python 代码执行后，其输出是（　　）。

```
cnt = 0
for i in range(10):
    cnt += 1
print(i, cnt)
```

- A. `10 10`
- B. `9 9`
- C. `10 9`
- D. `9 10`

第 9 题 下列 Python 代码执行后，其输出是（　　）。

```
cnt = 0
for i in range(-10, 0):
    cnt += 1
print(i * cnt)
```

- [ ] **A.** `-10`
- [ ] **B.** `-9`
- [ ] **C.** `0`
- [ ] **D.** `9`

第 10 题 下列 Python 代码执行后，其输出是（　　）。

```
for i in range(1,5):
    if i % 3 == 0:
        print(i, end = "+")
        continue
    else:
        print(i, end = "-")
        break
print(i, "END", end = "#")
```

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

第 11 题 如下 Python 代码执行时，下列说法**错误**的是（　　）。

```
N = int(input("请输入整数: "))
tot = 0

while N > 0:
    print(N, end = "#")
    tot += 1
    N -= 1

print(tot, "END", sep ="-")
```

- [ ] **A.** 如果输入 `3`，输出是：`3#2#1#3-END`
- [ ] **B.** 如果输入 `-10`，输出是：`0-END`
- [ ] **C.** 如果输入 `0`，输出是：`0-END`
- [ ] **D.** 如果输入 `1`，输出是：`1#0-END`

第 12 题 下列 Python 代码执行时如输入 `35`，输出如下图所示。横线处应填入的代码是（　　）。

```
 35 34 33 32 31 30 29 28 27 26
 25 24 23 22 21 20 19 18 17 16
 15 14 13 12 11 10  9  8  7  6
  5  4  3  2  1
```

```
N = int(input())

tot = 0
for i in range(N):
    print(__________, end = "")
    tot = tot + 1
    if tot % 10 == 0:
        print()
```

- [ ] **A.** `f"{N:3}"`
- [ ] **B.** `f"{N - i:3}"`
- [ ] **C.** `N - i`
- [ ] **D.** `N`

第 13 题 下列 Python 代码执行后，其画出的直线数量是（　　）。

```
import turtle

turtle.goto(100,100)
turtle.goto(100,0)
turtle.home()

turtle.done()
```

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

第 14 题 下列 Python 代码执行后，海龟朝向是（　　）。

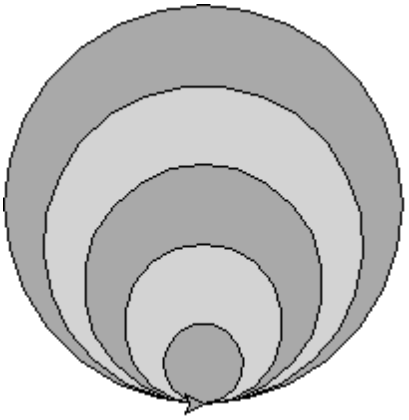
```
import turtle

turtle.left(90)
turtle.forward(100)
turtle.penup()
turtle.goto(0,0)

turtle.done()
```

- [ ] **A.** 向右
- [ ] **B.** 向上
- [ ] **C.** 向左
- [ ] **D.** 向下

第 15 题 要实现如下图所示的执行效果，横线应填入的代码是（　　）。



```python
import turtle

for i in range(____________):
    if i % 2 == 0:
        turtle.fillcolor("lightgray") #浅灰
    else:
        turtle.fillcolor("darkgray") #深灰

    turtle.begin_fill()
    turtle.circle(20 * i)
    turtle.end_fill()

turtle.done()
```

- ☐ **A.** `5`
- ☐ **B.** `1, 5`
- ☐ **C.** `5, 0, -1`
- ☐ **D.** `5, 0`

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 Python 程序执行 `input()` 函数时通常是使用键盘等输入设备来输入某个变量的值。

第 2 题 如果 `n` 为 3 位数的正整数，则 Python 表达式 `n % 100` 将能得到最高位。如 `n` 为 `123`，则该表达式的值为 `1`。

第 3 题 下面 Python 代码执行后将输出 `20`。

```python
for i in range(0, 20, 3):
    if i % 3 != 0:
        continue
print(i)
```

第 4 题 下列 Python 代码执行时如果先后输入 `10` 和 `10`，则将输出 `10小于等于10`。

```python
N = int(input())
M = int(input())

if N > M:
    print(f"{N}大于{M}")
else:
    print(f"{N}小于等于{M}")
```

第 5 题 如下 Python 代码如果输入大于 0 的整数，则可以求 1 到 `N` 之和。

```python
N = int(input("请输入正整数: "))
tot = 0

while N > 0:
    tot += N
    N += -1
print(tot)
```

第 6 题 在 Python 中，执行 `print( 10 > 9)` 后，其输出是 `TRUE` 。

第 7 题 在下列 Python 代码中，如果 `N` 和 `M` 均输入大于 0 的整数，则可以取得正整数 `N` 的从右到左第 `M` 位置的数。个位数为第 1 位置， `M` 不大于 `N` 的位数。

```python
N = int(input("请输入正整数:"))
M = int(input("从右到左第几位? "))

print(N // (10 ** (M - 1)) % 10)
```

第 8 题 求解鸡兔同笼。在如下 Python 代码中， `tou` 代表鸡和兔的头总数， `jiao` 代表鸡和兔的脚总数。先后两个输入均为正整数。该程序能求解鸡兔同笼。

```python
tou = int(input("请输入鸡和兔的头总数: "))
jiao = int(input("请输入鸡和兔的脚总数: "))

for tu_tou in range(tou + 1):
    ji_tou = tou - tu_tou
    if tu_tou * 4 + ji_tou * 2 == jiao:
        print(f"兔的数量={tu_tou},鸡的数量={ji_tou}")
if tu_tou == tou:
    print("无解")
```

第 9 题 在 Python `turtle` 中， `turtle.right(90)` 与 `turtle.left(-90)` 的效果相同。

第 10 题 下列 Python 代码执行后，将产生只有两条互相垂直的直线。

```python
import turtle

turtle.goto(-50,0)
turtle.goto(0,50)
turtle.goto(50,0)
turtle.goto(0,-50)

turtle.done()
```

3　编程题（每题 25 分，共 50 分）

3.1　编程题 1

- **试题名称**：新龟兔赛跑
- **时间限制**：1.0 s
- **内存限制**：512.0 MB

3.1.1　题目描述

自从上次龟兔赛跑，兔子因为轻敌而惨败后，一直心有不甘，因此有了新一届龟兔赛跑。

乌龟和上次一样，将会以恒定的速度往终点爬行，题目将给出乌龟从起点到终点所需时间；
兔子并没有吸取教训，只是打算在终点前少睡一会儿，题目将给出兔子的速度，和兔子打算在终点前睡觉的时间；

起点和终点之间的距离固定为 1000.0，请帮忙计算乌龟和兔子谁会获胜？

3.1.2　输入格式

输入包含 3 行，每行一个浮点数：

- 第 1 行的浮点数表示乌龟从起点到终点所需时间；
- 第 2 行的浮点数表示兔子的速度；
- 第 3 行的浮点数表示兔子打算在终点前睡觉的时间。

3.1.3　输出格式

输出包含 2 行：

- 第 1 行，如果乌龟获胜，输出 `turtle`；如果兔子获胜，输出 `rabbit`；如果平局，输出 `tie`；
- 第 2 行，输出兔子到达终点所需的总时间（含兔子睡觉的时间），保留两位小数。

3.1.4　样例

3.1.5　输入样例 1

```
20.0
100.0
11.0
```

3.1.6　输出样例 1

```
turtle
21.00
```

3.1.7　样例解释 1

兔子跑完全程所需时间是 $1000.0 \div 100.0 = 10.0$，而兔子要睡觉 11.0 单位时间，所以兔子一共需要 21.0 单位时间才能到达终点，这大于乌龟所需时间。

3.1.8　输入样例 2

```
20.0
100.0
9.0
```

3.1.9 输出样例 2

```
rabbit
19.00
```

3.1.10 样例解释 2

与样例 1 类似，兔子跑完全程所需时间是 $1000.0 \div 100.0 = 10.0$，但兔子仅睡觉 9.0 单位时间，所以兔子一共需要 19.0 单位时间就能到达终点，这小于乌龟所需时间。

3.1.11 数据范围

所有输入均有且仅有一位小数，且是不超过 1000.0 的正数。

本题默认给出的数值均为龟兔世界中的标准单位，不需要进行任何单位换算。

路程 = 速度 × 时间

3.1.12 参考程序

```python
# 接收乌龟到终点的时间，兔子速度，兔子的睡眠时间
t1 = float(input())
v = float(input())
t2 = float(input())

# 兔子到达的时间，1000/v是跑步的时间，t2是睡觉的时间
tt = 1000.0 / v + t2
# 乌龟到达时间大于兔子到达时间
if tt < t1:
    print('rabbit')
# 乌龟到达时间小于兔子到达时间
elif tt > t1:
    print('turtle')
# 平局
else:
    print('tie')
print("%.02f" % tt)
```

3.2 编程题 2

- **试题名称**：棋盘上的奖赏
- **时间限制**：1.0 s
- **内存限制**：512.0 MB

3.2.1 题目描述

相传古印度舍罕王为了奖赏宰相西萨发明国际象棋，西萨提的任何要求舍罕王都愿意满足。西萨只求在棋盘第 1 格放 1 粒麦子，第 2 格放 2 粒麦子，之后每一格都放前一格两倍的麦子，以此类推直到第 64 格。国王觉得这奖赏微不足道，立刻命人取粮。

然而麦子的数量飞速增长：到第 20 格已需上百万粒，到第 40 格就耗尽了全国库存，而后面仍有 24 格。最终国王发现，即便倾尽天下粮食也填不满棋盘。

聪明的你知道前 n 格总共需要多少粒麦子吗?

3.2.2 输入格式

一行，一个整数 n。

3.2.3 输出格式

一行，一个整数，表示前 n 格总共需要的麦子粒数。

3.2.4 样例

3.2.5 输入样例 1

```
3
```

3.2.6 输出样例 1

```
7
```

3.2.7 输入样例 2

```
10
```

3.2.8 输出样例 2

```
1023
```

3.2.9 数据范围

对于 40% 的测试点，保证 $1 \leq n \leq 10$。

对于所有测试点，保证 $1 \leq n \leq 30$。

3.2.10 参考程序

```python
n = int(input())
# f:当前的格子有多少麦子，ans: 总共有多少麦子
f, ans = 1, 0
# 遍历棋盘的每一个格子
for i in range(n):
    # 把当前格子的麦子加入到总结果
    ans += f
    # 下一个格子的麦子数目乘以2
    f *= 2
print(ans)
```