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

# Python 四级

2025 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 | A | A | C | D | C | B | A | A | D | C | B | D | B | D |

**第 1 题** 2025年4月19日在北京举行了一场颇为瞩目的人形机器人半程马拉松赛。比赛期间，跑动着的机器人会利用身上安装的多个传感器所反馈的数据来调整姿态、保持平衡等，那么这类传感器类似于计算机的( )。

- **A.** 处理器
- **B.** 存储器
- **C.** 输入设备
- **D.** 输出设备

**第 2 题** 小杨购置的计算机使用一年后觉得内存不够用了，想购置一个容量更大的内存条，这时需要的内存条是（）。

- **A.** RAM
- **B.** ROM
- **C.** CACHE
- **D.** EPROM

**第 3 题** 执行下面Python代码后，输出的结果是？（）

```
s1 = {('a', 1), ('b', 2)}
s2 = {('b', 2), ('a', 1)}
t1 = (('a', 1), ('b', 2))
t2 = (('b', 2), ('a', 1))
print(s1 == s2, t1 == t2)
```

- **A.** True False
- **B.** False False
- **C.** True True
- **D.** False True

**第 4 题** 执行下面Python代码后，输出的结果是？（）

```python
def func(*args):
    return sum(args) / len(args)


result = func(1, 2, 3, 4, 5)
print(result)
```

- [ ] **A.** 15
- [ ] **B.** 3
- [ ] **C.** 3.0
- [ ] **D.** 报错

**第 5 题** 以下哪个函数调用是合法的？（ ）

```python
def func(a, b, *, c, d=0):
    pass
```

- [ ] **A.** `func(1, 2, 3, 4)`
- [ ] **B.** `func(a=1, 2, c=3)`
- [ ] **C.** `func(1, 2, 3, d=4)`
- [ ] **D.** `func(1, 2, c=3)`

**第 6 题** 执行下面Python代码后，输出的结果是？（ ）

```python
x = 10


def func():
    x += 1
    return x


print(func())
```

- [ ] **A.** 10
- [ ] **B.** 11
- [ ] **C.** 报错
- [ ] **D.** None

**第 7 题** 执行下面Python代码后，输出的结果是？（ ）

```python
func = lambda x: x * 2 if x > 0 else x // 2
print(func(-4), func(4))
```

- [ ] **A.** 2 -8
- [ ] **B.** -2 8
- [ ] **C.** -8 2
- [ ] **D.** 报错

**第 8 题** 执行下面Python代码后，输出的结果是？（ ）

```python
def apply(func, x):
    return func(x)

def square(n):
    return n * n

result = apply(square, 4)
print(result)
```

- [ ] **A.** 16
- [ ] **B.** 8
- [ ] **C.** 4
- [ ] **D.** 报错

**第 9 题** 执行下面Python代码后，会发生什么？（ ）

```python
try:
    raise ValueError
except Exception:
    print("A")
except ValueError:
    print("B")
```

- [ ] **A.** 打印"A"
- [ ] **B.** 打印"B"
- [ ] **C.** 同时打印"A"和"B"
- [ ] **D.** 报错

**第 10 题** 执行下面Python代码后，输出的结果是？（ ）

```python
try:
    1 / 0
except ValueError:
    print("A")
else:
    print("B")
print("C")
```

- [ ] **A.** A C
- [ ] **B.** B C
- [ ] **C.** C
- [ ] **D.** 报错

**第 11 题** 以下哪个选项可以正确读取文件"data.txt"的全部内容并返回一个字符串？

```python
with open("data.txt", "r") as f:
    content = ________
```

- [ ] **A.** `f.readlines()`

- [ ] **B.** `f.readline()`
- [ ] **C.** `f.read()`
- [ ] **D.** `f.read(100)`

**第 12 题** 以下哪种文件打开模式可以在不截断文件的情况下同时支持读写？（ ）

- [ ] **A.** "w"
- [ ] **B.** "r+"
- [ ] **C.** "a"
- [ ] **D.** "x"

**第 13 题** 以下哪种排序算法的时间复杂度在最坏情况下是$O(n^2)$？（ ）

- [ ] **A.** 冒泡排序
- [ ] **B.** 选择排序
- [ ] **C.** 插入排序
- [ ] **D.** 以上都是

**第 14 题** 执行以下代码后，输出的结果是？（ ）

```python
data = ['apple', 'Banana', 'cherry']
print(sorted(data))
```

- [ ] **A.** `['apple', 'Banana', 'cherry']`
- [ ] **B.** `['Banana', 'apple', 'cherry']`
- [ ] **C.** `['apple', 'cherry', 'Banana']`
- [ ] **D.** `['cherry', 'Banana', 'apple']`

**第 15 题** 给定一个整数数组 `nums`，计算其最长递增子序列的长度。子序列可以不连续，但必须保持原数组的顺序。例如：`nums = [10, 9, 2, 5, 3, 7, 101, 18]` 的最长递增子序列是 `[2, 3, 7, 101]`，长度为 4。

```python
def length_of_LIS(nums):
    dp = [1] * len(nums)  # dp[i] 表示以 nums[i] 结尾的最长递增子序列长度
    for i in range(1, len(nums)):
        for j in range(i):
            if nums[j] < nums[i]:
                dp[i] = ________  # 填空
    return max(dp)
```

- [ ] **A.** `dp[j] + 1`
- [ ] **B.** `dp[i] + 1`
- [ ] **C.** `min(dp[i], dp[j] + 1)`
- [ ] **D.** `max(dp[i], dp[j] + 1)`

## 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** 现在，人们参加各种闭卷考试时通常都不允许将智能手机、平板电脑等带入考场，因为智能手表通常都有嵌入操作系统及通信等功能，所以也不允许携带入考场。（ ）

**第 2 题** 执行下面Python代码后，输出的结果为3。

```
data = {'ids': [1, 2], 'name': 'test'}
data['ids'].extend(['g', 'e', 's', 'p'])
print(len(data['ids']))
```

**第 3 题** 执行下面Python代码会报错，因为lambda函数不能有默认参数。

```
f = lambda x, y=2: x + y
f(1)
```

**第 4 题** 执行下面Python代码后，输出的结果为 `[1, 4, 9]` 。

```
print(list(map(lambda x: x**2, [1, 2, 3])))
```

**第 5 题** 执行下面Python代码后，将只输出 `try` 。

```
def test():
    try:
        return "try"
    finally:
        print("finally执行了")

print(test())
```

**第 6 题** 在Python中， `readlines()` 方法返回的文件内容包含每行末尾的换行符（ `\n` ）。

**第 7 题** 递推算法的核心思想是利用已知条件逐步推导后续结果。

**第 8 题** 选择排序是一种稳定的排序算法。

**第 9 题** 以下代码的时间复杂度是 `O(n)` 。

```
def func(n):
    for i in range(n):
        for j in range(n):
            print(i, j)
```

**第 10 题** 以下代码的空间复杂度是 `O(1)` 。

```
def sum_elements(arr):
    total = 0
    for num in arr:
        total += num
    return total
```

# 3　编程题（每题 25 分，共 50 分）

## 3.1　编程题 1

- **试题名称**：画布裁剪
- **时间限制**：3.0 s
- **内存限制**：512.0 MB

### 3.1.1　题目描述

小 A 在高为 $h$ 宽为 $w$ 的矩形画布上绘制了一幅画。由于画布边缘留白太多，小 A 想适当地裁剪画布，只保留画的主体。具体来说，画布可以视为 $h$ 行 $w$ 列的字符矩阵，其中的字符均为 ASCII 码位于 $33 \sim 126$ 之间的可见字符，小 A 只保留画布中由第 $x_1$ 行到第 $x_2$ 行、第 $y_1$ 列到第 $y_2$ 列构成的子矩阵。

小 A 将画布交给了你，你能帮他完成画布的裁剪吗?

### 3.1.2　输入格式

第一行，两个正整数 $h, w$，分别表示画布的行数与列数。

第二行，四个正整数 $x_1, x_2, y_1, y_2$，表示保留的行列边界。

接下来 $h$ 行，每行一个长度为 $w$ 的字符串，表示画布内容。

### 3.1.3　输出格式

输出共 $x_2 - x_1 + 1$ 行，每行一个长度为 $y_2 - y_1 + 1$ 的字符串，表示裁剪后的画布。

### 3.1.4　样例

#### 3.1.4.1　输入样例 1

```
3 5
2 2 2 4
.....
.>_<.
.....
```

#### 3.1.4.2　输出样例 1

```
>_<
```

#### 3.1.4.3　输入样例 2

```
5 5
1 2 3 4
AbCdE
fGhIk
LmNoP
qRsTu
VwXyZ
```

#### 3.1.4.4 输出样例 2

```
Cd
hI
```

### 3.1.5 数据范围

对于所有测试点，保证 $1 \leq h, w \leq 100$，$1 \leq x_1 \leq x_2 \leq h$，$1 \leq y_1 \leq y_2 \leq w$。

### 3.1.6 参考程序

```python
n, m = map(int, input().split())
x1, x2, y1, y2 = map(int, input().split())

s = [input().strip() for _ in range(n)]
# 遍历需要输出的行
for i in range(x1 - 1, x2):
    # 对需要输出的内容进行切片
    print(s[i][y1 - 1 : y2])
```

## 3.2 编程题 2

- **试题名称**：排序
- **时间限制**：3.0 s
- **内存限制**：512.0 MB

### 3.2.1 题目描述

体育课上有 $n$ 名同学排成一队，从前往后数第 $i$ 位同学的身高为 $h_i$，体重为 $w_i$。目前排成的队伍看起来参差不齐，老师希望同学们能按照身高从高到低的顺序排队，如果身高相同则按照体重从重到轻排序。在调整队伍时，每次只能交换相邻两位同学的位置。老师想知道，最少需要多少次交换操作，才能将队伍调整成目标顺序。

### 3.2.2 输入格式

第一行，一个正整数 $n$，表示队伍人数。

接下来 $n$ 行，每行两个正整数 $h_i$ 和 $w_i$，分别表示第 $i$ 位同学的身高和体重。

### 3.2.3 输出格式

输出一行，一个整数，表示最少需要的交换次数。

### 3.2.4 样例

#### 3.2.4.1 输入样例 1

```
5
1 60
3 70
2 80
4 55
4 50
```

#### 3.2.4.2 输出样例 1

```
8
```

#### 3.2.4.3 输入样例 2

```
5
4 0
4 0
2 0
3 0
1 0
```

#### 3.2.4.4 输出样例 2

```
1
```

### 3.2.5 数据范围

对于所有测试点，保证 $1 \leq n \leq 3000$，$0 \leq h_i, w_i \leq 10^9$。

### 3.2.6 参考程序

```python
n = int(input().strip())
a = []
for _ in range(n):
    h, w = map(int, input().split())
    a.append((h, w))

# 统计最终的交换次数
ans = 0
# 统计是否还需要继续排序
flag = True
while flag:
    flag = False
    i = 0
    while i < n - 1:
        if a[i] < a[i + 1]:
            # 只要发生过一次交换，就还需要继续排序
            flag = True
            a[i], a[i + 1] = a[i + 1], a[i]
            ans += 1
            i += 1
print(ans)
```