

全国青少年信息素养大赛-2026-初中组-算法-Py-丝路-复赛-模拟题-01-答案与解析

1	2	3	4	5	6	7	8	9	10
B	A	A	B	B	A、B、 C、D	A、B、 D	A、 B、 C、D	A、 B、 C、D	A、 B、D

一、单选题

(1) 丝路贸易公司在复制驼队货物清单（嵌套列表）时使用了 `copy()` 方法。执行以下代码后，输出结果为（ ）。

```
1 a = [1, 2, [3, 4]]
2 b = a.copy()
3 b[2][0] = 99
4 print(a[2][0])
```

- A. 3
- B. 99
- C. [3, 4]
- D. 0

【正确答案】 B

【题目解析】

✔ 正确答案

B (99)

💡 核心考点

浅拷贝 `copy()` 不复制嵌套对象。

🔍 执行追踪

`b` 是 `a` 的浅拷贝，内层列表 `[3,4]` 仍与 `a` 共享引用，修改 `b[2][0]` 同步影响 `a[2][0]`。

⚠ 易错提醒

浅拷贝只复制最外层，深拷贝需使用 `copy.deepcopy()`。

(2) 丝路关税司比对两批过境商队编号，需要找出两批共有的编号（交集）和全部出现过的编号（并集）。执行以下代码后，输出结果为（ ）。

```
1 a = {1, 2, 3}
2 b = {2, 3, 4}
3 print(a & b)
4 print(a | b)
```

- A. {2, 3} 和 {1, 2, 3, 4}
- B. {1, 4} 和 {1, 2, 3, 4}
- C. {2, 3} 和 {1, 2, 3, 4, 2, 3}
- D. {1, 2, 3, 4} 和 {2, 3}

【正确答案】 A

【题目解析】

✅ 正确答案

A

💡 核心考点

集合的交集 `&` 与并集 `|` 运算。

🔍 执行追踪

`a & b` 取共同元素得 `{2, 3}`；`a | b` 合并去重得 `{1, 2, 3, 4}`。

⚠ 易错提醒

集合自动去重，并集不会重复出现元素。

(3) 丝路物流港按货运费用公式为各商队计算运费，支持通过关键字指定参数。执行以下代码后，输出结果为（ ）。

```
1 def calc(x, y=2):
2     return x ** y
3
4 print(calc(y=3, x=2))
```

- A. 8
- B. 6
- C. 9
- D. 程序会报错

【正确答案】 A

【题目解析】

✓ 正确答案

A (8)

💡 核心考点

关键字参数传递。

🔍 执行追踪

x=2, y=3, 计算 $2^3 = 8$ 。

⚠ 易错提醒

关键字参数可以打乱顺序传入，不影响赋值。

(4) 丝路商贸中心统计了各口岸通关评分，需要按评分降序排列找出排名第一的口岸。执行以下代码后，输出结果为 () 。

```
1 scores = [('霍尔果斯', 90), ('阿拉山口', 85), ('二连浩特', 95)]
2 scores.sort(key=lambda x: x[1], reverse=True)
3 print(scores[0])
```

- A. ('霍尔果斯', 90)
- B. ('二连浩特', 95)
- C. ('阿拉山口', 85)
- D. 95

【正确答案】 B

【题目解析】

✓ 正确答案

B ('二连浩特', 95)

💡 核心考点

考察 sort() 配合 lambda 按元组指定元素降序排序。

🔍 执行追踪

按每个元组第 2 个元素降序排序后，列表变为 `[('二连浩特', 95), ('霍尔果斯', 90), ('阿拉山口', 85)]`，取首元素。

⚠️ 易错提醒

`reverse=True` 表示降序；输出的是整个元组而非分数。

(5) 丝路物流园用函数同时返回货运统计中的最小值和最大值。执行以下代码后，输出结果为（ ）。

```
1 def min_max(nums):
2     return min(nums), max(nums)
3
4 result = min_max([5, 2, 8, 1])
5 print(type(result))
6 print(result[0])
```

- A. `<class 'list'>` 和 1
- B. `<class 'tuple'>` 和 1
- C. `<class 'tuple'>` 和 5
- D. `<class 'list'>` 和 5

【正确答案】 B

【题目解析】

✅ 正确答案

B

💡 核心考点

函数返回多个值时默认打包为元组。

🔍 执行追踪

`min_max` 返回 `(1, 8)`，类型为 `tuple`，`result[0]` 为 1。

⚠️ 易错提醒

用逗号分隔的多返回值是元组（tuple），不是列表（list）。

二、多选题

(6) 丝路贸易公司在复制跨境物流档案时，需要理解深浅拷贝的区别。关于 Python 的深浅拷贝，以下说法正确的有（ ）。

- A. `copy()` 是浅拷贝，只复制最外层对象
- B. 浅拷贝后的嵌套列表与原列表共享内部的子列表
- C. 使用 `deepcopy()` 可以创建完全独立的副本（需要 `import copy`）
- D. 对不可变对象（如整数）组成的列表，浅拷贝和深拷贝效果相同

【正确答案】 A、B、C、D

【题目解析】

✅ 正确答案

A、B、C、D

💡 核心考点

浅拷贝与深拷贝的区别。

🔍 执行追踪

`copy()` 仅复制最外层，嵌套子列表仍为引用共享；`deepcopy()` 递归复制所有层级，需 `import copy`；不可变对象不存在修改问题，两者效果等同。

⚠️ 易错提醒

浅拷贝对嵌套结构修改子列表会影响原列表，这是最常见的坑。

(7) 丝路驿站用递归算法生成商队通行密码的排列。关于递归函数，以下说法正确的有（ ）。

- A. 递归函数必须至少有一个终止条件（基准情况）
- B. 没有终止条件的递归会导致栈溢出
- C. 递归函数不能有返回值
- D. 任何递归问题都可以用循环来等价实现

【正确答案】 A、B、D

【题目解析】

✅ 正确答案

A、B、D

💡 核心考点

递归函数的基本性质与特征。

🔍 执行追踪

A 正确，递归必须有基准情况以终止；B 正确，无终止条件会无限调用导致栈溢出；C 错误，递归函数可以有返回值；D 正确，递归与迭代在理论上是等价的，任何递归都可改写为循环（可能需要借助栈）。

⚠ 易错提醒

不要误以为递归函数不能返回值，实际上大多数递归函数都通过返回值传递结果。

(8) 丝路海关在解密跨境密电过程中使用完善的异常处理结构。关于 `try-except-else-finally` 异常处理结构，以下说法正确的有（ ）。

- A. `else` 子句只在 `try` 块没有发生异常时执行
- B. `finally` 子句无论是否发生异常都会执行
- C. `except` 可以有多个，用于捕获不同类型的异常
- D. `else` 必须出现在所有 `except` 之后、`finally` 之前

【正确答案】 A、B、C、D

【题目解析】

✅ 正确答案

A、B、C、D

💡 核心考点

考察 `try-except-else-finally` 异常处理结构的语法与执行顺序。

🔍 执行追踪

`try` 正常→执行 `else`；无论异常与否→执行 `finally`；多个 `except` 按顺序匹配；语法顺序固定为 `try→except→else→finally`。

⚠ 易错提醒

`else` 只在无异常时触发，且必须位于所有 `except` 之后、`finally` 之前，顺序不可颠倒。

(9) 丝路档案馆需要将通关记录写入文件并妥善保管。关于文件操作，以下说法正确的有（ ）。

- A. `'r'` 模式表示以只读方式打开文件
- B. `'w'` 模式会覆盖文件原有内容
- C. `'a'` 模式表示在文件末尾追加内容
- D. 使用 `with` 语句打开文件，可以自动关闭文件

【正确答案】 A、B、C、D

【题目解析】

✅ 正确答案

A、B、C、D

💡 核心考点

Python 文件打开模式与 `with` 语句的用法。

🔍 执行追踪

`'r'` 只读、`'w'` 写入并覆盖、`'a'` 末尾追加，`with` 语句会在代码块结束后自动调用 `close` 关闭文件，四项描述均正确。

⚠️ 易错提醒

注意 `'w'` 模式会清空原文件内容，若需保留旧数据应使用 `'a'` 模式。

(10) 丝路驼队在丝路沿线搜索最优商旅路线。关于搜索算法，以下说法正确的有 ()。

- A. DFS（深度优先搜索）使用栈或递归实现
- B. BFS（广度优先搜索）使用队列实现
- C. DFS 一定能找到最短路径
- D. 在网格搜索中，通常使用方向数组（如 `dx = [0,0,1,-1]`，`dy = [1,-1,0,0]`）来遍历四个方向的相邻位置

【正确答案】A、B、D

【题目解析】

✅ 正确答案

A、B、D

*1. 🧠 解题思路

本题考查搜索算法（DFS 与 BFS）的基本特性及网格搜索的常用技巧。

*2. ⚙️ 逻辑推演

- A 正确：DFS 沿一条路径深入到底再回溯，天然适合用递归实现，递归本质上利用了系统调用栈；也可以显式使用栈来模拟。
- B 正确：BFS 按层扩展，需要先进先出的数据结构，使用队列（queue）实现。
- C 错误：DFS 找到的路径不一定最短，它只保证能找到一条可行路径。求最短路径在无权图中通常使用 BFS。
- D 正确：方向数组是网格搜索中的常用技巧，通过 `dx`、`dy` 数组配合循环即可简洁地遍历上下左右四个相邻格子，避免重复书写四段判断代码。

*3. ✅ 参考代码

```

1  # 网格搜索中方向数组的典型用法
2  dx = [0, 0, 1, -1]
3  dy = [1, -1, 0, 0]
4  x, y = 2, 3 # 当前位置
5  for i in range(4):
6      nx, ny = x + dx[i], y + dy[i] # 计算四个相邻位置
7      # 接下来判断 (nx, ny) 是否合法并继续搜索

```

*4. ⚠️ 警示与分析

易错点：

误以为 DFS 也能找最短路径。实际上 DFS 是"一条路走到黑"，找到的第一条路径未必最短；BFS 在无向图中才能保证最短。

混淆 DFS 与 BFS 的数据结构：DFS 用栈（或递归），BFS 用队列，切勿弄反。

三、编程题

(11) Y3694 丝路驿站货仓

题目描述

丝路驿站货仓负责管理沿线商队的货物存取。假设对于任意一种贸易商品，每天开始营业时的库存总量已知，并且一天之内不会通过进货的方式增加。每天会有多支商队前来提货，每支商队希望提走不同数量的货物。如果商队需要的数量超过了当时的库存量，货仓就会拒绝该商队的请求。

请你编写程序：输入商品初始库存、商队数量及每支商队的需求量，计算有多少支商队没有提到货物。

输入描述

共 3 行：

- 第一行是每天开始时的商品库存总量 m ；
- 第二行是这一天提货的商队数量 n （ $0 < n \leq 100$ ） ；
- 第三行共有 n 个数，分别记录了每支商队希望提走的货物数量（按照先后顺序）。

输出描述

一行，一个整数，表示没有提到货物的商队数量。

输入样例 1

```

1  30
2  6
3  10 5 20 6 7 8

```

输出样例 1

【正确答案】见解析

【题目解析】

✅ 解题思路

本题是经典的贪心模拟问题。按照提货顺序依次处理，维护剩余库存，当库存不足时计数拒绝数量。

💡 核心考点

贪心算法模拟、累加与判断。

🔍 执行追踪

依次处理每支商队：库存30→商队1取10剩20→商队2取5剩15→商队3需20但剩15，拒绝（计数1）→商队4取6剩9→商队5取7剩2→商队6需8但剩2，拒绝（计数2），共拒绝2支。

⚠️ 易错提醒

注意是"当前库存不足时才拒绝"，不是累计需求超过总量就拒绝。

```

1  m = int(input())
2  n = int(input())
3  requests = list(map(int, input().split()))
4
5  rejected = 0
6  remaining = m
7
8  for req in requests:
9      if req <= remaining:
10         remaining -= req
11     else:
12         rejected += 1
13
14  print(rejected)

```

(12) Y3695 丝路密码拼图

题目描述

丝路文化展使用火柴棒在展厅中摆出数字密码，向观众展示丝绸之路沿线的古城编号。用火柴棒拼数字

0 - 9 的拼法为：0 需 6 根、1 需 2 根、2 需 5 根、3 需 5 根、4 需 4 根、5 需 5 根、6 需 6 根、7 需 3 根、8 需 7 根、9 需 6 根。

现在有 n 根火柴棒，丝路文化传承人要用它们拼出形如 $A+B=C$ 的等式密码。注意：

1. 加号和等号各自需要 2 根火柴棒；

2. 如果 A 不等于 B , 则 $A+B=C$ 或 $B+A=C$ 视为不同的等式 (A 、 B 、 $C \geq 0$) ;

3. n 根火柴棒必须全部用上。

请你编写程序, 计算能拼成的不同等式的数目。

输入描述

一行, 有一个整数 n ($1 \leq n \leq 24$) 。

输出描述

一行, 有一个整数, 表示能拼成的不同等式的数目。

输入样例 1

1 14

输入样例 2

1 18

输出样例 1

1 2

输出样例 2

1 9

【正确答案】见解析

【题目解析】

✅ 解题思路

本题是枚举+哈希表的综合题。先预处理每个数字 (0~11111范围) 所需火柴棒数量, 然后枚举A和B, 计算C, 检查火柴棒总数是否恰好为n。

💡 核心考点

枚举法、哈希表预处理。

🔍 执行追踪

火柴棒消耗表: 0→6, 1→2, 2→5, 3→5, 4→4, 5→5, 6→6, 7→3, 8→7, 9→6。

对于n=14: 去掉+和=各2根, 剩10根给A、B、C。枚举A、B, 检查

$\text{match}(A) + \text{match}(B) + \text{match}(A+B) = 10$ 的方案数。

⚠ 易错提醒

注意A!=B时A+B=C和B+A=C算两个不同等式; A==B时只算一个。数字0也需要6根火柴棒。

```

1  def match_count(x):
2      """计算数字x需要的火柴棒数量"""
3      matches = [6, 2, 5, 5, 4, 5, 6, 3, 7, 6]
4      if x == 0:
5          return 6
6      count = 0
7      while x > 0:
8          count += matches[x % 10]
9          x //= 10
10     return count
11
12     n = int(input())
13     # 加号和等号共需要 2+2=4 根火柴棒
14     remain = n - 4
15     ans = 0
16
17     # 枚举 A 和 B, 数字范围不会太大 (最多 1111 左右)
18     for A in range(0, 1200):
19         for B in range(0, 1200):
20             C = A + B
21             if match_count(A) + match_count(B) + match_count(C) == remain:
22                 if A != B:
23                     ans += 2 # A+B=C 和 B+A=C 算不同
24                 else:
25                     ans += 1
26
27     print(ans)

```

(13) Y3696 丝路商队名录解密

题目描述

小英是丝路贸易学院跨境物流专业的学员，暑假期间获得了去中欧班列调度中心实习的机会。主任交给她一项任务——解密丝绸之路上被加密过的一些商队名录。经过研究，小英发现了如下加密规律（括号中是一个“原文 → 密文”的例子）：

1. 原文中所有的字符都在字母表中被循环左移了三个位置（dec → abz）。
2. 逆序存储（abcd → dcba）。
3. 大小写反转（abXY → ABxy）。

请你编写程序：输入加密后的字符串，输出解密后的商队名录。

输入描述

一个加密的字符串。（长度小于 50 且只包含大小写字母）

输出描述

输出解密后的字符串。

输入样例 1

```
1 GS00WFAS0q
```

输出样例 1

```
1 Trvdizrrvj
```

【正确答案】见解析

【题目解析】

✅ 解题思路

本题是字符串处理题。解密是加密的逆过程：先逆序（还原逆序）→再循环右移3位（还原左移）→最后大小写反转（还原大小写反转）。

💡 核心考点

字符串处理、循环移位、逆序、大小写转换。

🔍 执行追踪

加密过程：原文→左移3位→逆序→大小写反转=密文。

解密过程：密文→大小写反转（还原）→逆序（还原）→右移3位（还原）。

以GS00WFAS0q为例：大小写反转→gsoowfasoQ→逆序→QOasfwoosg→右移3位→Trvdizrrvj。

⚠️ 易错提醒

注意解密顺序要与加密顺序严格相反。循环右移时注意边界（z→a→b→c的循环）。

```

1  s = input()
2
3  # 步骤1: 大小写反转
4  step1 = ''
5  for ch in s:
6      if ch.isupper():
7          step1 += ch.lower()
8      else:
9          step1 += ch.upper()
10
11 # 步骤2: 逆序
12 step2 = step1[::-1]
13
14 # 步骤3: 循环右移3位 (还原左移3位)
15 result = ''
16 for ch in step2:
17     if 'a' <= ch <= 'z':
18         result += chr((ord(ch) - ord('a') + 3) % 26 + ord('a'))
19     else:
20         result += chr((ord(ch) - ord('A') + 3) % 26 + ord('A'))
21
22 print(result)

```

(14) Y3697 丝路商道寻路

题目描述

丝路商队在复杂的古商道路网中行进时，需要穿越一片多变区域。该区域可以看成是由 $n \times n$ 的格点组成，每个格点只有 2 种状态，`.` 和 `#`，前者表示可通行的商道，后者表示无法通行的荒漠。商队只能向上下左右四个方向之一移动。商队需要从 A 点赶到 B 点会合，问在不走出区域的情况下能不能办到。如果起点或者终点在荒漠中（为 `#`），则无法办到。

输入描述

第 1 行是测试数据的组数 k ，后面跟着 k 组输入。每组测试数据的第 1 行是一个正整数 n ($1 \leq n \leq 100$)，表示区域的规模是 $n \times n$ 的。接下来是一个 $n \times n$ 的矩阵，矩阵中的元素为 `.` 或者 `#`。再接下来一行是 4 个整数 ha 、 la 、 hb 、 lb ，描述 A 处在第 ha 行、第 la 列， B 处在第 hb 行、第 lb 列。注意 ha 、 la 、 hb 、 lb 全部是从 0 开始计数的。

输出描述

k 行，每行输出对应一个输入。能办到则输出 `YES`，否则输出 `NO`。

输入样例 1

```
1  2
2  3
3  .##
4  ..#
5  #..
6  0 0 2 2
7  5
8  .....
9  ###.#
10 ..#..
11 ###..
12 ...#.
13 0 0 4 0
```

输出样例 1

```
1  YES
2  NO
```

【正确答案】 见解析

【题目解析】

✅ 解题思路

本题是标准的网格BFS/DFS连通性判断问题。使用BFS或DFS从起点出发搜索是否能到达终点。

💡 核心考点

BFS/DFS网格搜索、连通性判断。

🔍 执行追踪

从起点(ha,la)开始BFS/DFS，标记访问过的格子，如果遇到终点(hb,lb)则输出YES，搜索结束后未到达则输出NO。

⚠️ 易错提醒

注意先检查起点和终点是否为#，如果是直接输出NO。网格坐标从0开始计数。BFS用队列，DFS用递归或栈。

```

1  from collections import deque
2
3  def bfs(grid, n, ha, la, hb, lb):
4      if grid[ha][la] == '#' or grid[hb][lb] == '#':
5          return False
6
7      visited = [[False] * n for _ in range(n)]
8      queue = deque()
9      queue.append((ha, la))
10     visited[ha][la] = True
11     dx = [0, 0, 1, -1]
12     dy = [1, -1, 0, 0]
13
14     while queue:
15         x, y = queue.popleft()
16         if x == hb and y == lb:
17             return True
18         for i in range(4):
19             nx, ny = x + dx[i], y + dy[i]
20             if 0 <= nx < n and 0 <= ny < n and not visited[nx][ny] and grid[nx][ny] == '.':
21                 visited[nx][ny] = True
22                 queue.append((nx, ny))
23     return False
24
25     k = int(input())
26     for _ in range(k):
27         n = int(input())
28         grid = []
29         for _ in range(n):
30             grid.append(input().strip())
31         ha, la, hb, lb = map(int, input().split())
32         if bfs(grid, n, ha, la, hb, lb):
33             print("YES")
34         else:
35             print("NO")

```