

青少年人工智能编程水平测试七级试题

一、单项选择题（每题 2 分，共 15 题，共 30 分）

1. 在 Python 中，哪种数据类型不适合用于表示多维数组？

- A. 列表
- B. 字典
- C. 集合
- D. 元组

正确答案：C

解析：集合（set）是无序的，不适合用于表示多维数组。

2. 以下哪个选项正确创建了一个包含三个元素的元组？

- A. tuple3 = (1, 2, 3)
- B. tuple3 = [1, 2, 3]
- C. tuple3 = {(1, 2, 3)}
- D. tuple3 = {1: 2, 3: 4}

正确答案：A

解析：选项 A 创建了一个包含三个元素的元组。

3. 以下哪个 Python 内置函数不能作为高阶函数使用？

- A. map
- B. filter
- C. sum
- D. max

正确答案：C

解析：sum 函数不是高阶函数，不能接受另一个函数作为参数。

4. 关于 lambda 表达式，以下描述正确的是？

- A. lambda 表达式可以没有参数
- B. lambda 表达式可以有多个返回值
- C. lambda 表达式可以嵌套使用
- D. lambda 表达式必须包含 return 语句

正确答案：A

解析：lambda 表达式可以没有参数，也可以作为匿名函数使用。

5. 以下哪个 Python 表达式正确地使用了 filter 和 lambda 来筛选小于 10 的元素？

- A. result = list(filter(lambda x: x < 10, [7, 10, 15, 3]))
- B. result = map(lambda x: x < 10, [7, 10, 15, 3])
- C. result = [x for x in [7, 10, 15, 3] if x < 10]
- D. result = filter(10, [7, 10, 15, 3])

正确答案：A

解析：选项 A 使用了 filter 高阶函数和 lambda 表达式来筛选出小于 10 的元素。

6. 以下哪种方法不是 Pillow 库中的图像调整函数？

- A. rotate()
- B. flip()
- C. enhance()
- D. filter()

正确答案：D

解析：filter() 是 Python 内置的函数，不是 Pillow 库中的图像调整函数。

7. 使用 Pillow 库裁剪图像的函数是？

- A. crop()
- B. resize()
- C. cut()
- D. trim()

正确答案：A

解析：crop() 函数用于裁剪图像，指定裁剪区域的边界框。

8. 在 Pandas 中，哪个方法用于选择特定的行或列？

- A. select()
- B. filter()
- C. loc()
- D. iloc()

正确答案：C

解析：loc() 方法用于通过标签选择 DataFrame 中的行或列。

9. 在 Pandas 中，以下哪个选项创建了一个包含字符串的 Series？

- A. pd.Series(['a', 'b', 'c'])
- B. pd.Series([1, 2, 3])
- C. pd.Series({'a': 1, 'b': 2})
- D. pd.Series({1, 2, 3})

正确答案：A

解析：选项 A 创建了一个包含字符串元素的 Series。

10. 在 Pandas 中，以下哪个选项准确描述了 DataFrame 的特征？

- A. DataFrame 是一个多维数组
- B. DataFrame 是一个二维数据结构
- C. DataFrame 只能包含数值型数据
- D. DataFrame 的列名必须是字符串

正确答案：B

解析：DataFrame 是 Pandas 中的二维数据结构，可以包含多种类型的数据。

11. 以下哪个 HTTP 方法用于获取服务器上资源的描述？

- A. GET
- B. POST

C. HEAD

D. PUT

正确答案：C

解析：HEAD 方法用于获取资源的描述而不返回资源本身，通常用于检查资源是否存在。

12. 以下哪种排序算法的平均时间复杂度为 $O(n \log n)$ ？

A. 插入排序

B. 冒泡排序

C. 归并排序

D. 选择排序

正确答案：C

解析：归并排序的平均时间复杂度为 $O(n \log n)$ ，适用于大规模数据排序。

13. 在算法分析中， $O(n^2)$ 表示什么意思？

A. 算法的复杂度与数据的平方成正比

B. 算法的复杂度与数据的大小成正比

C. 算法的复杂度与数据的对数成正比

D. 算法的复杂度与数据的大小无关

正确答案：A

解析： $O(n^2)$ 表示算法的时间或空间复杂度与输入数据的平方成正比。

14. 以下哪种排序算法在最坏情况下的时间复杂度为 $O(n^2)$ ？

A. 归并排序

B. 快速排序

C. 堆排序

D. 插入排序

正确答案：D

解析：插入排序的最坏时间复杂度为 $O(n^2)$ ，适用于小规模数据排序。

15. 以下关于时间复杂度的描述中，哪一个是正确的？

A. 时间复杂度为 $O(n)$ 的算法在处理任意大小的数据集时，其执行时间总是与数据集的大小成正比

B. 时间复杂度为 $O(\log n)$ 的算法在输入规模较小时比 $O(n)$ 的算法更快

C. 时间复杂度为 $O(n^2)$ 的算法在输入规模增加时，其执行时间将会成倍增加

D. 时间复杂度为 $O(1)$ 的算法在某些情况下可能会比时间复杂度为 $O(n)$ 的算法更慢

正确答案：A

解析：B 选项与实际情况相反；C 选项描述不准确；D 选项是不可能的。

二、多项选择题（每题 3 分，共 5 题，共 15 分，多选或少选不得分）

1. 关于 Python 字典的说法中，哪些是正确的？

A. 字典中的键必须是不可变的

B. 字典中的值可以是任何数据类型

C. 字典是按插入顺序存储键值对的

D. 字典的键可以重复

正确答案: A, B, C

解析: A 正确: 字典中的键必须是不可变的类型, 如字符串、数字或元组。B 正确: 字典中的值可以是任何数据类型。C 正确: 从 Python 3.7 开始, 字典按照插入顺序存储键值对。D 错误: 字典中的键必须是唯一的, 不能重复。

2. 以下哪些是 Python 集合 (set) 的特性?

A. 集合中的元素是无序的

B. 集合中的元素不能重复

C. 集合支持索引操作

D. 集合可以用来进行数学集合操作, 如并集和交集

正确答案: A, B, D

解析: A 正确: 集合中的元素是无序的, 不能通过索引访问。B 正确: 集合中的元素不允许重复。C 错误: 集合不支持索引操作。D 正确: 集合支持数学集合操作, 如并集、交集、差集等。

3. 观察以下代码段, 选择描述正确的选项:

```
items = [2, 3, 4, 5, 6]
```

```
result = list(filter(lambda x: x % 2 != 0, map(lambda x: x + 1, items)))
```

A. 代码段使用了两个高阶函数

B. result 列表将包含所有 items 列表中的元素

C. map 函数用于将 items 列表中的每个元素加 1

D. result 列表将包含元素 [3, 5, 7]。

正确答案: A, C, D

解析: A 正确: 代码段使用了 map 和 filter 两个高阶函数。B 错误: result 列表中元素经过了 map 和 filter 处理, 不会包含原始列表的所有元素。C 正确: map 函数用 lambda 表达式为列表中每个元素加 1。D 正确: result 列表将包含 [3, 5, 7], 因为这些是经过操作后的奇数。

4. 观察以下代码, 选择正确的描述:

```
import pandas as pd
```

```
data = {'Product': ['A', 'B', 'C', 'D'],
```

```
        'Price': [100, 150, 200, 250],
```

```
        'Stock': [10, 20, 30, 40]}
```

```
df = pd.DataFrame(data)
```

```
product_price = df.loc[df['Price'] > 150, 'Product']
```

```
stock_result = df.iloc[1:3]
```

A. DataFrame 是通过字典创建的, 其中键作为列名

B. product_price 是一个 Series 对象, 包含价格大于 150 的产品名称

C. stock_result 是一个 DataFrame 对象, 包含第二行和第三行的数据

D. loc 和 iloc 均可用于选择 DataFrame 中的行和列。

正确答案: A, B, C, D

解析：A 正确：DataFrame 可以通过字典创建，字典键作为列名。B 正确：product_price 是通过条件选择得到的 Series 对象。C 正确：stock_result 通过 iloc[1:3] 获取，它选择了 DataFrame 的第二和第三行，并保持 DataFrame 对象的结构。D 正确：loc 和 iloc 都可以用于选择 DataFrame 中的行和列，只是用法稍有不同。

5. 使用 Pillow 库处理图像时，下列哪些操作是可以使用库中函数直接实现的？

- A. 将图像转换为 RGBA 模式
- B. 裁剪图像的特定区域
- C. 调整图像的大小
- D. 添加水印到图像上

正确答案：A, B, C

解析：A 正确：Pillow 支持将图像转换为 RGBA 模式。B 正确：可以使用 crop() 方法裁剪图像的特定区域。C 正确：可以使用 resize() 方法调整图像大小。D 错误：Pillow 库本身不直接支持添加水印，但可以通过合并图像的方式实现。

三、编程题（共 4 题，共 55 分）

注：试题均不允许在输入函数的括号中写入内容，输出函数仅输出题目要求的信息。所有程序运行时间限制为 3000MS，内存限制为 512MByte。

1. （本题共 5 组测试数据，每个测试点 2 分，共 10 分）

某城市计划在未来的 N 天内修建一段重要的道路。为了减少对城市交通的影响，施工队计划选择一个连续的 K 天时间段进行施工，使得这段时间内受影响的车辆数量最少。

输入描述

第 1 行有两个用空格隔开的整数，分别表示 N 和 K ($1 < K < N < 100$)。

第 2 行有 N 个用空格隔开的整数，依次表示接下来 N 天中每天通过该路段的车辆数量。

输出描述

一个整数，表示连续 K 天内受影响的最小车辆数量。

样例输入

10 4

300 200 400 600 150 250 500 100 200 350

样例输出

1000

示例题解程序：

```
def min_traffic_disruption(n, k, traffic):
    min_traffic_sum = float('inf')

    for i in range(n - k + 1):
        current_sum = sum(traffic[i:i+k])
        min_traffic_sum = min(min_traffic_sum, current_sum)

    return min_traffic_sum

# 读取第一行的 n 和 k
n, k = map(int, input().split())
# 读取第二行的车辆数量
traffic = list(map(int, input().split()))

# 计算并输出结果
result = min_traffic_disruption(n, k, traffic)
print(result)
```

2. （本题共 5 组测试数据，每个测试点 2 分，共 10 分）

你是一名自动化仓库的管理员，负责管理一个具有自动分拣系统的仓库。仓库内有一个传送带系统，负责将不同的物品运送到指定的分拣口。传送带上有多个分拣口，每个分拣口可以处理特定类型的物品。所有需分拣物品按顺序排列在一起在传送带上按照物品序号顺序移动，直到到达适合当前物品的分拣口处即被自动分拣。

每个物品都有一个唯一的编号和目标分拣口编号。传送带每移动一个位置需要花费 1 秒钟（例如哦那个分拣口 1 移动到分拣口 4 将花费 3 秒钟），分拣一个物品需要花费 2 秒钟。如果某个分拣口已经塞满了 10 件物品，系统将发出警报并停止操作。

请编写一个程序来模拟分拣任务的运行情况，计算总共花费的时间，并检测是否有任何分拣口出现溢出。

注意事项：

1. 每个分拣口的最大容量是 10 件物品。
2. 系统开机时所有物品在传送带上位于分拣口 1 的位置并自动从序号 1 的物品开始移动和分拣，随后不断继续。
3. 如果传送带在某一时刻出现分拣口溢出，系统将立即停止并输出警报。

输入描述：

第一行是一个整数 n ，表示物品的数量（100 到 200 之间）。

第二行包含 n 个整数，每个整数表示物品的目标分拣口编号（范围为 1 到 10）。

输出描述：

如果没有出现分拣口溢出，输出一个整数，表示完成所有分拣任务所需的总时间（单位为秒）。

如果某个分拣口出现溢出，输出“Overflow x ”，其中 x 是发生溢出的分拣口编号。

样例输入：

```
15
1 2 3 4 5 6 7 8 9 10 1 2 3 4 5
```

样例输出：

```
52
```


示例题解程序：

```
def simulate_sorting(n, targets):
    sorting_points = [0] * 10 # 初始化每个分拣口的计数器
    total_time = 0

    pos = 1 # 传送带上物品当前位置
    for i in range(n):
        target = targets[i] - 1 # 将目标分拣口编号转换为索引（0 到 9）
        sorting_points[target] += 1

        # 检查分拣口是否溢出
        if sorting_points[target] > 10:
            print(f"Overflow {target + 1}")
            return

        # 计算移动物品所花的时间
        total_time += abs(targets[i] - pos)

        # 分拣物品花费 2 秒
        total_time += 2

        # 重置当前位置
        pos = targets[i]

    # 如果没有溢出，输出总时间
    print(total_time)

# 读取输入
n = int(input())
targets = list(map(int, input().split()))

# 模拟分拣过程
simulate_sorting(n, targets)
```

3. （本题共 5 组测试数据，每个测试点 3 分，共 15 分）

小明非常喜欢植物，他在家养了一棵小树。这棵小树每隔一天会长出一个新分支，每个新分支在未来的每一天都可能继续长出新的分支。如果小明坚持给小树浇水，那么他想知道到第 n 天，这棵小树最多会有多少个分支。

假设第 1 天小树有 1 个主干分支。随后每一天，小树的每一个分支中的一半（向上取整）会再长出一个新分支。在这种情况下：

第 1 天：1 个分支（主干）

第 2 天：2 个分支（主干和主干上新长出的 1 个分支）

第 3 天：3 个分支（新长出 1 个分支）

第 4 天：5 个分支（新长出 2 个分支）

小明想知道，如果给小树连续浇水 n 天，最终这棵小树会有多少个分支。

输入描述：

一个正整数 n ($1 \leq n \leq 20$) 和一个正整数 m ，表示浇水的天数和初始的分支数量。

输出描述：

一个正整数，表示第 n 天小树有多少个分支。

样例输入：

4 1

样例输出：

5

示例题解程序（基本递归解法）：

```
import math

def count_branches(n):
    if n == 1:
        return m
    last_day = count_branches(n - 1)
    new_branches = math.ceil(last_day / 2)
    return last_day + new_branches

# 输入天数和初始分支数量
n, m = map(int, input().split())
# 输出第 n 天的分支的数量
print(count_branches(n))
```

4. (本题共 10 组测试数据, 每个测试点 2 分, 共 20 分)

在某个农场里, 有一种杂草具有极强的繁殖能力。每株杂草在出现后的第 5 天开始, 每天可以新生长出 2 株新杂草。新长出的杂草也会在它们的第 5 天开始, 每天长出 2 株新的杂草。

例如:

第 1 天, 农场里有 1 株杂草。

这株杂草从第 5 天开始, 每天会生长出 2 株新杂草。

因此, 第 5 天时杂草数量为 3 株。

第 6 天时数量为 5 株, 以此类推。

请问, 第 n 天时, 农场里共有多少株杂草?

输入格式:

一个正整数 n ($1 \leq n \leq 20$) 和一个正整数 m , 分别表示第几天和初始的杂草数量。

输出格式:

一个整数, 表示该天的杂草数。

样例输入

6 1

样例输出

5

示例题解程序（递归解法）：

```
def count_weeds(day, initial_weeds):  
    if day < 5:  
        return initial_weeds  
    return count_weeds(day - 1, initial_weeds) + 2 * count_weeds(day - 5,  
initial_weeds)  
  
# 输入第几天和初始的杂草数量  
n, m = map(int, input().split())  
# 输出该天的杂草数量  
print(count_weeds(n, m))
```