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

Python 四级

2024 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	A	C	C	D	A	B	A	A	C	D	C	C	D	B	A

第 1 题 据有关资料，山东大学于1972年研制成功DJL-1计算机，并于1973年投入运行，其综合性能居当时全国第三位。DJL-1计算机运算控制部分所使用的磁心存储元件由磁心颗粒组成，设计存贮周期为2μs（微秒）。那么该磁心存储元件相当于现代计算机的（ ）。

- ☐ **A.** 内存
- ☐ **B.** 磁盘
- ☐ **C.** CPU
- ☐ **D.** 显示器

第 2 题 IPv4版本的因特网总共有（ ）个A类地址网络。

- ☐ **A.** 65000
- ☐ **B.** 200万
- ☐ **C.** 126
- ☐ **D.** 128

第 3 题 执行下面Python代码后，输出的结果是？（ ）

```
my_list = [1, 2, [3, 4]]
my_list[2].extend([5, 6])
print(my_list)
```

- ☐ **A.** [1, 2, 5, 6]
- ☐ **B.** [1, 2, [3, 4]]
- ☐ **C.** [1, 2, [3, 4, 5, 6]]
- ☐ **D.** [1, 2, [3, 4, [5, 6]]]

第 4 题 执行下面Python代码后，会发生什么？（ ）

```
my_tuple = (1, 2, (3, 4))
my_tuple[2] = (5, 6)
```

- [ ] **A.** my_tuple变为(1, 2, (5, 6))
- [ ] **B.** 代码正常执行，没有变化
- [ ] **C.** 抛出ValueError异常
- [ ] **D.** 抛出TypeError异常

第 5 题 执行下面Python代码后，输出的结果是？（ ）

```python
try:
    result = 10 / int('a')
except ValueError:
    print("10", end="#")
else:
    print("20", end="#")
finally:
    print("30", end="#")
```

- [ ] **A.** 10#30#
- [ ] **B.** 10#
- [ ] **C.** 20#
- [ ] **D.** 30#

第 6 题 执行下面Python代码后，输出的结果是？（ ）

```python
def func(n):
    return len([num for num in range(n) if num % 2 == 0])


print(func(20))
```

- [ ] **A.** 8
- [ ] **B.** 10
- [ ] **C.** 11
- [ ] **D.** 15

第 7 题 执行下面Python代码后，输出的结果是？（ ）

```python
def func(*args):
    return ''.join(args)


print(func('Hello', 'World'))
```

- [ ] **A.** HelloWorld
- [ ] **B.** Hello World
- [ ] **C.** "Hello" "World"

☐ **D.** 抛出异常

第 8 题 执行下面Python代码后，输出的结果是？（ ）

```
def func(lst):
    lst.append(10)
    return lst

lstA = [1,2,3]

func(lstA)
print(lstA,func(lstA))
```

☐ **A.** [1, 2, 3, 10, 10] [1, 2, 3, 10, 10]

☐ **B.** [1, 2, 3] [1, 2, 3, 10]

☐ **C.** [1, 2, 3, 10] [1, 2, 3, 10]

☐ **D.** [1, 2, 3, 10] [1, 2, 3, 10, 10]

第 9 题 执行下面Python代码后，输出的结果是？（ ）

```
def tpADD(tpl):
    tpl = tpl +(5,6)
    return tpl

tp = (1,2,3)
tpADD(tp)
print(tp,tpADD(tp))
```

☐ **A.** (1, 2, 3, 5, 6, 5, 6) (1, 2, 3, 5, 6, 5, 6)

☐ **B.** (1, 2, 3, 5, 6) (1, 2, 3, 5, 6)

☐ **C.** (1, 2, 3) (1, 2, 3, 5, 6)

☐ **D.** (1, 2, 3) (1, 2, 3)

第 10 题 执行下面Python代码后，输出的结果是？（ ）

```
x = 5


def foo():
    def bar():
        global x
        x = 10

    bar()
    print(x, end="#")


foo()
print(x, end="#")
```

- [ ] **A.** 5#5#
- [ ] **B.** 10#5#
- [ ] **C.** 5#10#
- [ ] **D.** 10#10#

第 11 题 执行下面Python代码后，输出的结果是？（ ）

```python
def func(d):
    d['a'] = 5


c = {'a': 1, 'b': 2}
func(c)
print(c)
```

- [ ] **A.** {'a': 1, 'b': 2}
- [ ] **B.** {'b': 2}
- [ ] **C.** {'a': 5, 'b': 2}
- [ ] **D.** 抛出异常

第 12 题 以下Python代码实现的排序算法的时间复杂度是？（ ）

```python
def func_sort(arr):
    n = len(arr)
    for i in range(n - 1):
        min_idx = i
        for j in range(i+1, n):
            if arr[j] < arr[min_idx]:
                min_idx = j
        arr[i], arr[min_idx] = arr[min_idx], arr[i]
```

- [ ] **A.** O(n)
- [ ] **B.** O(2n)
- [ ] **C.** O(n^2)
- [ ] **D.** O(n^3)

第 13 题 执行下面Python代码后，输出的结果是？（ ）

```python
tuples = [(1, 'apple'), (2, 'banana'), (0, 'cherry')]
sorted_tuples = sorted(tuples, key=lambda x: x[1])
print(sorted_tuples)
```

- [ ] **A.** `[(1, 'apple'), (0, 'cherry'), (2, 'banana')]`
- [ ] **B.** `[(2, 'banana'), (1, 'apple'), (0, 'cherry')]`
- [ ] **C.** `[(0, 'cherry'), (1, 'apple'), (2, 'banana')]`

☐ **D.** `[(1, 'apple'), (2, 'banana'), (0, 'cherry')]`

第 14 题 假设你正在爬楼梯，每次可以爬1阶或2阶。给定楼梯的阶数 n ，计算有多少种不同的方法可以爬到楼顶。以下Python代码的横线处应该填写？（ ）

```python
def climbStairs(n):
    if n == 1:
        return 1
    if n == 2:
        return 2

    # 初始化前两阶楼梯的数据
    dp = [0] * (n + 1)
    dp[1] = 1
    dp[2] = 2

    # 从第3阶楼梯开始，计算每一阶楼梯的爬法数量
    for i in range(3, n + 1):
        _________
    return dp[n]
```

☐ **A.** dp[i] = 2 * dp[i - 1] + dp[i - 2]

☐ **B.** dp[i] = dp[i - 1] + dp[i - 2]

☐ **C.** dp[i] = 2 * dp[i - 2]

☐ **D.** dp[i] = dp[i - 1] + 2 * dp[i - 2]

第 15 题 文件numbers.txt的内容如下：

```
5
12
7
15
3
20
8
```

执行下面Python代码后，输出的结果是？（ ）

```python
def func(file_path, threshold):
    lst = []
    with open(file_path) as file:
        for line in file:
            number = int(line.strip())
            if number > threshold:
                lst.append(number)
    return lst


file_path = 'numbers.txt'
threshold = 12
selected_numbers = func(file_path, threshold)
print(selected_numbers)
```

- [ ] **A.** [15, 20]
- [ ] **B.** [12, 15, 20, 8]
- [ ] **C.** [5, 12, 7, 15, 20, 8]
- [ ] **D.** [12, 15, 20]

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 小杨最近开始学习C++编程，老师说C++是一门面向对象的编程语言,也是一门高级语言。（ ）

第 2 题 在Python程序中，自定义函数可以定义在主程序代码的前面，也可以定义在主程序代码的后面，都不会发生错误。（ ）

第 3 题 在Python程序中，如果自定义函数内没有return语句或者return语句不带任何返回值，那么该函数的返回值为False。

第 4 题 try-except-else-finally异常处理结构中，只有try程序段中的语句没有异常，else程序段中的语句才会得到执行。

第 5 题 Python中避免使用反斜线 `\` 指定文件路径时出错，如 `C:\test\numbers.txt`；常常使用正斜线 `/` 或者双反斜线 `\\`。

第 6 题 'w'可以作为open()函数的参数，表示以写的方式打开文件，若文件不存在，则会抛出异常。

第 7 题 下面这段程序的时间复杂度为线性阶$O(n)$。（ ）

```
def func(n):
    i = 0
    while i ** 2 < n:
        i += 1
```

第 8 题 对一组数据 [5, 2, 6, 4, 8, 1, 7, 3]使用冒泡的方法按从大到小的顺序进行排序，则第2轮排序过后的结果是[6, 5, 8, 4, 7, 3, 2, 1]。

第 9 题 执行下面Python代码后，输出的结果为(0, 0)。

```
lst = [(x, x**2) for x in range(-5, 6)]
a = min(lst, key=lambda x: abs(x[1]))
print(a)
```

第 10 题 在Python中表达式 `{1, 3, 5} & {2, 4, 6} == {}` 的值为True。

3　编程题（每题 25 分，共 50 分）

3.1　编程题 1

- **试题名称**：黑白方块
- **时间限制**：1.0 s
- **内存限制**：512.0 MB

3.1.1 题面描述

小杨有一个 n 行 m 列的网格图，其中每个格子要么是白色，要么是黑色。

小杨想知道网格图中是否存在一个满足如下条件的子矩形：

- 子矩形由 4 行 4 列组成；
- 子矩形的第 1 行 和第 4 行只包含白色格子；
- 对于子矩形的第 2 行 和第 3 行，只有第 1 个和第 4 个格子是白色的，其余格子都是黑色的；

请你编写程序帮助小杨判断。

3.1.2 输入格式

第一行包含一个正整数 t，代表测试用例组数。

接下来是 t 组测试用例。对于每组测试用例，一共 $n+1$ 行。

第一行包含两个正整数 n, m，含义如题面所示。

之后 n 行，每行一个长度为 m 的 01 串，代表网格图第 i 行格子的颜色，如果为 0，则对应格子为白色，否则为黑色。

3.1.3 输出格式

对于每组测试用例，如果存在，输出 Yes，否则输出 No。

3.1.4 样例1

```
3
1 4
0110
5 5
00000
01100
01100
00001
01100
5 5
00000
01100
01110
00001
01100
```

```
No
Yes
No
```

满足条件的子矩形形如：

```
0000
0110
0110
0000
```

对于全部数据，保证有 $1 \leq t \leq 10, 1 \leq n, m \leq 100$。

3.1.5 参考程序

```python
N = 110
w = [[0] * N for _ in range(N)]
match = [[0] * 4 for _ in range(4)]
n, m = 0, 0

def check(xa, ya):
    for i in range(4):
        for j in range(4):
            if w[xa + i][ya + j] != match[i][j]:
                return False
    return True

def main():
    for i in range(1, 3):
        match[1][i] = match[2][i] = 1

    t = int(input())
    for _ in range(t):
        global n, m
        n, m = map(int, input().split())

        for i in range(1, n + 1):
            s = input()
            for j in range(1, m + 1):
                w[i][j] = int(s[j - 1])

        fl = False
        for i in range(1, n - 2):
            for j in range(1, m - 2):
                if check(i, j):
                    fl = True
                    break
            if fl:
                break

        if fl:
            print("Yes")
        else:
            print("No")

if __name__ == "__main__":
    main()
```

3.2 编程题 2

- **试题名称**：区间排序
- **时间限制**：1.0 s
- **内存限制**：512.0 MB

3.2.1 题面描述

小杨有一个包含 n 个正整数的序列 a。

小杨计划对序列进行多次升序排序，每次升序排序小杨会选择一个区间 $[l,r]$（$l \leq r$）并对区间内所有数字，即 $a_l, a_{l+1}, \ldots, a_r$ 进行升序排序。每次升序排序会在上一次升序排序的结果上进行。

小杨想请你计算出多次升序排序后的序列。

3.2.2 输入格式

第一行包含一个正整数 n，含义如题面所示。

第二行包含 n 个正整数 $a_1, a_2, \ldots, a_n$，代表序列。

第三行包含一个正整数 q，代表排序次数。

之后 q 行，每行包含两个正整数 l_i，r_i，代表将区间 $[l_i, r_i]$ 内所有数字进行升序排序。

3.2.3 输出格式

输出一行包含 n 个正整数，代表多次升序排序后的序列。

3.2.4 样例1

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

```
1 3 4 5 2
```

- 第一次升序排序后，序列为 $[3,4,5,1,2]$；
- 第二次升序排序后，序列为 $[3,4,1,5,2]$；
- 第三次升序排序后，序列为 $[1,3,4,5,2]$；

对于全部数据，保证有 $1 \leq n \leq 100, 1 \leq a_i \leq 100, 1 \leq q \leq 100, 1 \leq l_i \leq r_i \leq n$。

3.2.5 参考程序

```python
N = 1010
a = [0] * N
n = 0

def bubbleSort(l, r):
    flag = True
    while flag:
        flag = False
        for i in range(l, r):
            if a[i] > a[i + 1]:
                flag = True
                a[i], a[i + 1] = a[i + 1], a[i]

n = int(input())
```

```
s = input().split()
for i in range(1, n + 1):
    a[i] = int(s[i-1])

q = int(input())
while q:
    q -= 1
    l, r = map(int, input().split())
    bubbleSort(l, r)

for i in range(1, n + 1):
    print(a[i], end=" " if i != n else "\n")
```