python 七级

2023 年 12 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	B	D	C	C	D	D	B	C	B	C	D	C	B	D	B

第 1 题 假设变量 x 为 float 类型，如果下面代码输入为 100，输出最接近(　)。

```
import math
def function():
    x=float(input("Please input a:"))
    b=math.log10(x)-math.log2(x)
    print(b)
function()
```

☐ A.0

☐ B.-5

☐ C.-8

☐ D.8

第 2 题 对于下面动态规划方法实现的函数，以下选项中最适合表达其状态转移函数的为(　)。

```
#函数调用前，已定义s为list成员数量为MAX_N，
#f同为list，成员数量为MAX_N，每个成员为MAX_N个成员list
def stone_merge(n,a):
    for i in range(1,n+1):
        s[i] = s[i - 1] + a[i]
    for i in range(1,n+1):
        for j in range(1,n+1):
            if i == j:
                f[i][j] = 0
            else:
                f[i][j] = MAX_F #预定义的值
    for l in range(1, n):
        for i in range(1, n - l + 1):
            j = i + l
            for k in range(i,j):
                f[i][j] = min([f[i][j],f[i][k] + f[k + 1][j] + s[j] - s[i-1]])
    return f[1][n]
```

A. $f(i,j) = \min_{i \le k < j}(f(i,j), f(i,k) + f(k+1,j) + s(j) - s(i-1))$

B. $f(i,j) = \min_{i \le k < j}(f(i,j), f(i,k) + f(k+1,j) + \sum_{k=i}^{j} a(k))$

C. $f(i,j)=\min_{i\le k\le j}(f(i,k)+f(k+1,j)+\sum_{k=i}^{j+1}a(k))$

D. $f(i,j)=\min_{i\le k<j}(f(i,k)+f(k+1,j))+\sum_{k=i}^{j}a(k)$

第 3 题 下面代码可以用来求最长上升子序列（LIS）的长度，如果输入是整数序列是：5 1 7 3 5 9，则输出是(　)。

```
arr = list(map(int,input().split(",")))
cnt = 1
ans = 0

for i in range(len(arr)-1):
    if arr[i] < arr[i+1]:
        cnt += 1
    else:
        ans = max(ans,cnt)
        cnt = 1
    ans = max(ans,cnt)
    print(ans)
```

A. 9　7　5　1　1　9

B. 1　2　2　2　3

C. 1　3　5　7　9　9

D. 1　1　1　1　1　1

第 4 题 Python 中，下列关于类描述不正确的是(　)。

A. 类属性可以通过类的名称访问其值。

B. 类属性也可以称之为类的静态属性。

C. 和实例属性相同，类属性前面也必须有 self 关键字，并以句点间隔。

D. 如果在类的某个实例（对象）中修改类属性的值，则其他该类实例访问该值时，其值也随之改变。

第 5 题 G 是一个非连通无向图，共有 28 条边，则该图至少有(　)个顶点。

A. 6

B. 7

C. 8

D. 9

第 6 题 哈希表长 31，按照下面的程序依次输入 4 17 28 30 4，则 4 存入哪个位置？（ ）

```
N = 31
htab,flag = [0 for i in range(N)], [0 for i in range(N)]

n, x, i, j, k = 0, 0, 0, 0, 0

n = int(input())
for i in range(n):
    x = int(input())
    k = x % 13

    while(flag[k]):
        k = (k+1) % 13

    htab[k] = x
    flag[k] = 1

for i in range(N):
    print(htab[i],end="  ")
```

A. 3

B. 4

C. 5

D. 6

第 7 题 某二叉树 T 的先序遍历序列为：{A B D F C E G H}，中序遍历序列为：{B F D A G E H C}，则下列说法中正确的是(　)。

A. T 的度为 1

B. T 的高为 4

C. T 有 4 个叶节点

D. 以上说法都不对

第 8 题 下面代码段可以求两个字符串 s1 和 s2 的最长公共子串（LCS），下列相关描述不正确的是（　）。

```
def LCS(str1,str2):
    res = [[0]*len(str2) for _ in range(len(str1))]
    max_len=0
    max_str=[]
    for i in range(len(str1)):
        for j in range(len(str2)):
            if str1[i]==str2[j]:
                if i>0 and j>0:
                    res[i][j]=1+res[i-1][j-1]
                else:
                    res[i][j]=1
            if res[i][j]>max_len:
                max_len=res[i][j]
                max_str=[str1[i-max_len+1:i+1]]
            elif res[i][j]==max_len:
                max_str.append(str1[i-max_len:i])
    return max_str
str1,str2=(input().split())
n=LCS(str1,str2)
```

A. 代码的时间复杂度为 O(n^2)

B. 代码的空间复杂度为 O(n^2)

C. 空间复杂度已经最优

D. 采用了动态规划求解

第 9 题 图的广度优先搜索中既要维护一个标志数组标志已访问的图的结点，还需哪种结构存放结点以实现遍历？（ ）

A. 双向栈

B. 队列

C. 哈希表

D. 堆

第 10 题 对关键字序列{44，36，23，35，52，73，90，58}建立哈希表，哈希函数为 h(k)=k%7，执行下面的 Insert 函数，则等概率情况下的平均成功查找长度（即查找成功时的关键字比较次数的均值）为( )。

```
class Node:
    def __init__(self,Data=None,Next=None):
        self.Data = Data
        self.Next = Next
hTab = [Node() for i in range(7)]
Key = [44, 36, 23,35, 52, 73, 90, 58, 0]

def Insert():
    i, j = 0, 0
    x = Node()
    while Key[i]:
        j = Key[i] % 7
        x = Node();
        x.Data = Key[i]
        x.Next = hTab[j]
        hTab[j] = x

        i += 1

Insert()
```

A. 7/8

B. 1

C. 1.5

D. 2

第 11 题 学生在读期间所上的某些课程中需要先上其他的课程，所有课程和课程间的先修关系构成一个有向图 G，有向边<U, V>表示课程 U 是课程 V 的先修课，则要找到某门课程 C 的全部先修课下面哪种方法不可行？（ ）

A. BFS 搜索

B. DFS 搜索

C. DFS+BFS

D. 动态规划

第 12 题 一棵完全二叉树有 2023 个结点，则叶结点有多少个？（ ）

A. 1024

B. 1013

C. 1012

D. 1011

第 13 题 用下面的邻接表结构保存一个有向图 G，InfoType 和 VertexType 是定义好的类。设 G 有 n 个顶点、e 条弧，则求图 G 中某个顶点 u（其顶点序号为 k）的度的算法复杂度是(　)。

```
class ArcNode:
    def __init__(self,adjvex=None,nextarc=None,info=None):
        self.adjvex = adjvex
        self.nextarc = nextarc
        self.info = info

class VNode:
    def __init__(self,data=None,firstarc=None):
        self.data = data
        self.firstarc = firstarc

class ALGraph:
    def __init__(self,vertices,vexnum,arcnum,kind):
        self.vertices = vertices
        self.vexnum, self.arcnum = vexnum, arcnum
        self.kind = kind

AdjList = [VNode() for i in range(MAX_VERTEX_NUM])
```

A. O(n)

B. O(e)

C. O(n+e)

D. O(n+2*e)

第 14 题 给定一个简单有向图 G，判断其中是否存在环路的下列说法哪个最准确？(　)

A. BFS 更快

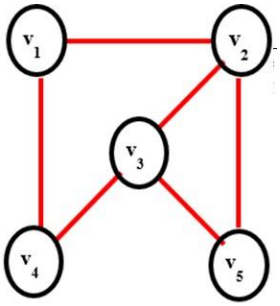
B. DFS 更快

C. BFS 和 DFS 一样快

D. 不确定

第 15 题 从顶点 v1 开始遍历下图 G 得到顶点访问序列，在下面所给的 4 个序列中符合广度优先的序列有几个？(　)

{v1 v2 v3 v4 v5} ，　{v1 v2 v4 v3 v5} ，　{v1 v4 v2 v3 v5} ，　{v1 v2 v4 v5 v3}



A. 4

B. 3

C. 2

D. 1

B

2 判断题（每题 2 分，共 20 分）

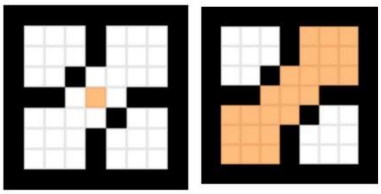
题号	1	2	3	4	5	6	7	8	9	10
答案	√	√	×	√	√	×	×	×	√	×

第 1 题　小杨这学期准备参加 GESP 的 7 级考试，其中有关于三角函数的内容，他能够通过下面的代码找到结束循环的角度值。（ ）

```
import math
def function():
    while True:
        x=float(input())
        x=x/180*3.14
        if int(math.sin(x)*math.sin(x)+math.cos(x)*math.cos(x))==1:
            print(x)
            break

function()
```

第 2 题 小杨在开发画笔刷小程序（applet），操作之一是选中黄颜色，然后在下面的左图的中间区域双击后，就变成了右图。这个操作可以用图的泛洪算法来实现。（ ）



第 3 题 假设一棵完全二叉树共有 N 个节点，则树的深度为 log(N)+1。（ ）

第 4 题 给定一个数字序列 A1，A2，A3，...，An，要求 i 和 j（1<=i<=j<=n），使 Ai+...+Aj 最大，可以使用动态规划方法来求解。（ ）

第 5 题 若变量 x 为 float 类型正数，则 log(math.exp(x))>math.log10(x)。（ ）

第 6 题　简单有向图有 n 个顶点和 e 条弧，可以用邻接矩阵或邻接表来存储，二者求节点 u 的度的时间复杂度一样。（ ）

第 7 题 某个哈希表键值 x 为整数，为其定义哈希函数 H(x)=x%p，则 p 选择素数时不会产生冲突。（ ）

第 8 题 动态规划只要推导出状态转移方程，就可以写出递归程序来求出最优解。（ ）

第 9 题　广度优先搜索（BFS）能够判断图是否连通。（ ）

第 10 题　在 Python 中，如果定义了构造函数，则创建对象时先执行完缺省的构造函数，再执行这个定义的构造函数。（ ）

3 编程题（每题 25 分，共 50 分）

3 编程题 (每题 25 分，共 50 分)

3.1 编程题 1

- **试题名称**：商品交易
- **时间限制**：1.0 s
- **内存限制**：128.0 MB

3.1.1 问题描述

市场上共有 N 种商品，编号从 0至N-1，其中，第 i 种商品价值 v_i 元。

现在共有 M 个商人，编号从 0至M-1。在第 j 个商人这，你可以使用第 x_j 种商品交换第 y_j 种商品。每个商人都会按照商品价值进行交易，具体来说，如果$v_{x_j} > v_{y_j}$，他将会付给你 v_{xj}>v_{yj}元钱；否则，那么你需要付给商人 V_{xj}-v_{yj}元钱。除此之外，每次交易商人还会收取 1 元作为手续费，不论交易商品的价值孰高孰低。

你现在拥有商品 a，并希望通过一些交换来获得商品 b。请问你至少要花费多少钱？（当然，这个最小花费也可能是负数，这表示你可以在完成目标的同时赚取一些钱。）

3.1.2 输入描述

第一行四个整数 N，M，a，b，分别表示商品的数量、商人的数量、你持有的商品以及你希望获得的商品。保证 0 ≤ α, b < N，保证 $a \neq b$。

第二行N个用单个空格隔开的正整数 $v_0, V_1, \ldots, v_{N\_1}$，依次表示每种商品的价值。保证 1 ≤ vi ≤ 10^9。

接下来 M行，每行两个整数 x_j,y_j，表示第 j 个商人愿意使用第 x_j 种商品交换第 y_j 种商品。保证 0≤x_j，y_j < N，保证 $x_j \neq y_j$。

3.1.3 输出描述

输出一行一个整数，表示最少的花费。特别地，如果无法通过交换换取商品 b，请输出 `No solution`

3.1.4 特别提醒

在常规程序中，输入、输出时提供提示是好习惯。但在本场考试中，由于系统限定，请不要在输入、输出中附带任何提示信息。

3.1.5 样例输入 1

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

3.1.6 样例输出 1

```
5
```

3.1.7 样例解释 1

可以先找 2 号商人，花 2 - 1 = 1 元的差价以及 1 元手续费换得商品 1，再找 4 号商人，花 4 - 2 = 2 元的差价以及 1 元手续费换得商品 2 。总计花费 1 + 1 + 2 + 1 = 5 元。

3.1.8 样例输入 2

```
3 3 0 2
100 2 4
0 1
1 2
0 2
```

3.1.9 样例输出 2

```
-95
```

3.1.10 样例解释 2

可以找 2 号商人，直接换得商品 2 的同时，赚取 100-4 = 96 元差价，再支付 1 元手续费，净赚 95 元。

也可以先找 0 号商人换取商品 1，再找 1 号商人换取商品 2，不过这样只能赚 94 元。

3.1.11 样例输入 3

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

3.1.12 样例输出 3

```
No solution
```

3.1.13 数据规模

对于30%的测试点，保证 $N \leq 10$，$M \leq 20$。

对于70%的测试点，保证 $N \leq 10^3$，$M \leq 10^4$。

对于100%的测试点，保证 $N \leq 10^5$，$M \leq 2 \times 10^5$

3.1.14 参考程序

```
from collections import deque

max_n = int(1e5) + 10
n = 0
edge = [[] for _ in range(max_n)]
val = [0] * max_n
min_dist = [float('inf')] * max_n
queue = deque()

def bfs(src):
    global min_dist
    queue.clear()
    min_dist = [float('inf')] * max_n
```

```
    queue.append(src)
    min_dist[src] = 0
    while queue:
        u = queue.popleft()
        for v in edge[u]:
            if min_dist[u] + 1 < min_dist[v]:
                min_dist[v] = min_dist[u] + 1
                queue.append(v)

if __name__ == "__main__":
    n, m, src, dst = map(int, input().split())
    n_values = list(map(int, input().split()))
    for i in range(n):
        val[i] = n_values[i]
    for _ in range(m):
        x, y = map(int, input().split())
        edge[x].append(y)

    bfs(src)
    if min_dist[dst] > n:
        print("No solution")
    else:
        print(min_dist[dst] - val[src] + val[dst])
```

3.2 编程题 2

- **试题名称**：纸牌游戏
- **时间限制**：1.0 s
- **内存限制**：128.0 MB

3.2.1 问题描述

你和小杨在玩一个纸牌游戏。

你和小杨各有 3 张牌，分别是 0、1、2。你们要进行N 轮游戏，每轮游戏双方都要出一张牌，并按 1 战胜 0、2 战胜 1，0 战胜 2 的规则决出胜负。第 i 轮的胜者可以获得 $2a_i$ 分，败者不得分，如果双方出牌相同，则算平局，二人都可获得 a_i 分（$i = 1, 2, \ldots, N$）。

玩了一会后，你们觉得这样太过于单调，于是双方给自己制定了不同的新规则。小杨会在整局游戏开始前确定自己全部 n 轮的出牌，并将他的全部计划告诉你；而你从第 2 轮开始，要么继续出上一轮出的牌，要么记一次“换牌”。游戏结束时，你换了 t 次牌，就要额外扣 b1 + ... + bt 分。请计算出你最多能获得多少分。

3.2.2 输入描述

第一行一个整数 N，表示游戏轮数。

第二行 N 个用单个空格隔开的非负整数 $\alpha_1, \ldots\ldots, \alpha_N$，意义见题目描述。

第三行 N-1 个用单个空格隔开的非负整数 $b_1, \ldots\ldots, b_{N-1}$，表示换牌的罚分，具体含义见题目描述。由于游戏进行 N 轮，所以你至多可以换 N - 1 次牌。

第四行 N 个用单个空格隔开的整数 C1,...,CN，依次表示小杨从第 1 轮至第 N 轮出的牌。保证 ci∈{0,1,2}。

3.2.3 输出描述

一行一个整数，表示你最多获得的分数。

3.2.4 特别提醒

在常规程序中，输入、输出时提供提示是好习惯。但在本场考试中，由于系统限定，请不要在输入、输出中附带任何提示信息。

3.2.5 样例输入 1

```
4
1 2 10 100
1 100 1
1 1 2 0
```

3.2.6 样例输出 1

```
219
```

3.2.7 样例解释 1

你可以第 1 轮出 0，并在第 2，3 轮保持不变，如此输掉第 1，2 轮，但在第 3 轮中取胜，获得 2 × 10 = 20 分；随后，你可以在第 4 轮中以扣 1 分为代价改出 1，并在第 4 轮中取得胜利，获得 2 × 100 = 200 分。如此，你可以获得最高的总分 20 + 200-1=219。

3.2.8 样例输入 2

```
6
3 7 2 8 9 4
1 3 9 27 81
0 1 2 1 2 0
```

3.2.9 样例输出 2

```
56
```

3.2.10 数据规模

对于30%的测试点，保证 $N \leq 15$。

对于60%的测试点，保证 $N \leq 100$。

对于所有测试点，保证 $N \leq 1,000$；保证 $0 \leq a_i, b_i \leq 10^6$。

3.2.11 参考程序

```
max_n = 1005

n = int(input())
a = [0] * max_n
b = [0] * max_n
c = [0] * max_n

dp = [[0] * max_n for _ in range(3)]


def result(x, y):
    if x == y + 1 or x == y - 2:
        return 2
    if x == y:
        return 1
    return 0


a[1:] = list(map(int, input().split()))
b[1:] = list(map(int, input().split()))
c[1:] = list(map(int, input().split()))

for k in range(3):
    dp[k][0] = result(k, c[1]) * a[1]

for i in range(2, n + 1):
    for j in range(i - 1, -1, -1):
        for k in range(3):
            curr_score = result(k, c[i]) * a[i]
            dp[k][j] += curr_score
            if j > 0:
                for l in range(3):
                    dp[k][j] = max(dp[k][j], dp[l][j - 1] + curr_score - b[j])

ans = -2e9
x = 0
for j in range(n):
    for k in range(3):
        ans = max(ans, dp[k][j])

print(ans)
```