

## 附件2：U15 组样题及说明(Python)

BBCC比赛按流程分为：初赛——网络赛、决赛——现场团队赛、现场个人赛；  
U15 组是面向年龄在 2020年 9 月 1 日前未满 16 周岁的青少年设置的比赛组别；  
此组别的测试题说明与题目样例如下：

### 1 试题说明

1 试题 U15 – A 用于 U15 组的网络个人赛，试题按类型分为：选择题、填空题、程序完善题和程序设计题，其中程序设计题评测方式与 NOI 类似，对于每道题目，选手只能提交一次代码，以此代码作为统一评测的答卷，测试时为每道题提供了 10 – 20 组测试数据，根据数据点判断得分。

2 试题 U15 – B 用于 U15 组的现场团队赛，考核题目为程序设计题，题目评测方式与 ICPC 类似，对于每道题目，每支队伍可以多次提交，采取实时反馈与罚时机制；

3 试题 U15 – C 用于 U15 组的现场个人赛，考核题目为程序设计题，评测方式与 IOI 类似，对于每道题目，每位选手可以多次提交，采取实时反馈与得分机制；

### 2 题目样例

#### U15 – A 选择题（样例）

##### 题目

在二进制下， $1011001 + (\quad) = 1100110$ 。

- A. 1011
- B. 1101
- C. 1010
- D. 1111

##### 解题思路

二进制正整数的加法问题，直接用减法计算即可。

##### 参考答案

1 | B

## U15 – A 填空题 ( 样例 )

### 题目

书架上有 10 本书，编号从 1 到 10，从其中选 3 本，其中每两本的编号都不相邻的选法一共有( ) 种。

### 解题思路

设从 10 个本书中任意抽走 3 本，剩下 7 本，抽走的书，应该原本的位置，应该在这 7 本书之间的 8 个间隔才符合要求，即可以将模型转换为：将 3 本书放在 8 个位置中的组合数，所以答案是  $C_8^3$ 。

### 参考答案

1 | 56

## U15 – A 程序完善题 ( 样例 )

### 题目

给输入一个  $n_1 \times m_1$  的矩阵  $a$ ，和  $n_2 \times m_2$  的矩阵  $b$ ，问  $a$  中是否存在子矩阵和  $b$  相等。若存在，输出所有子矩阵左上角的坐标；若不存在输出 “There is no answer”。

```
1 import numpy as np
2
3 n1 = int(input("n1=?"))
4 m1 = int(input("m1=?"))
5 a = np.zeros((n1, m1))
6 for i in range(n1):
7     for j in range(m1):
8         a[i][j] = int(input(f"a[{{i+1}}][{{j+1}}]=?"))
9
10 n2 = int(input("n2=?"))
11 m2 = int(input("m2=?"))
12 b = np.zeros((n2, m2))
13 for i in range(n2):
14     for j in range(m2):
15         (1)_____ = int(input(f"b[{{i+1}}][{{j+1}}]=?"))
16
17 haveAns = False
18
19 for i in range(n1-n2+1):
20     for j in range((2)_____):
21         (3)_____
22         for k1 in range(n2):
23             for k2 in range((4)_____):
24                 if (a[i+k1][j+k2] != b[k1][k2]):
25                     good = False
26             if good:
27                 print(f"i, j= {i+1}, {j+1}") # 加1映射问题中从1开始计数的坐标
28                 (5)_____
29
30 if not haveAns:
31     print("There is no answer")
```

### 参考答案

```
1 | (1) b[i][j]
2 | (2) m1-m2+1
3 | (3) good = True
4 | (4) m2
5 | (5) haveAns = true
```

## U15 程序设计题（样例）

### 神奇的电梯

#### 题目描述

猪猪学校有一个很奇怪的电梯，在大楼的每一层都可以停靠，而且第  $i$  层楼上有一个数字  $K_i$ 。电梯只有四个按钮：开，关，上，下。上下的层数等于当前楼层上的那个数字。当然，如果不能满足要求，相应的按钮就会失灵。例如：3, 3, 1, 2, 5 代表了  $K_i$  ( $K_1 = 3, K_2 = 3, \dots$ )，从 1 楼开始。在 1 楼，按“上”可以到 4 楼，按“下”是不起作用的，因为没有 -2 楼。那么，从  $A$  楼到  $B$  楼至少要按几次按钮呢？

#### 输入格式

共二行：

第一行为 3 个用空格隔开的正整数，表示  $N, A, B$ 。

第二行为  $N$  个用空格隔开的非负整数，表示  $K_i$ 。

#### 输出格式

一行，即最少按键次数，若无法到达，则输出 -1。

#### 样例输入

```
1 | 5 1 5
2 | 3 3 1 2 5
```

#### 样例输出

```
1 | 3
```

#### 样例分析

如上所述。

#### 数据范围

100% 的数据： $1 \leq N \leq 200, 1 \leq A, B \leq N$ ；

## 解题思路

本题可以用搜索的办法解决，对于每层电梯，都可以新增两个状态，但是状态可能重复，所以注意记忆化处理。

## 参考代码

```
1  from collections import deque
2
3  def bfs(n, st, ed, a):
4      l = deque()
5      l.append((st, 0))
6      f = [0] * n
7      f[st] = 1
8      while l:
9          kx, kt = l.popleft()
10         x = kx + a[kx] # 上楼
11         y = kx - a[kx] # 下楼
12         if (x == ed) or (y == ed):
13             return kt + 1
14         if (x >= 0) and (x < n) and (f[x] == 0):
15             l.append((x, kt+1))
16             f[x] = 1
17         if (y >= 0) and (y < n) and (f[y] == 0):
18             l.append((y, kt+1))
19             f[y] = 1
20     return -1
21
22 n = int(input("n=?"))
23 st = int(input("start=?"))
24 ed = int(input("end=?"))
25
26 msg = "k[1], k[2], ... k[n]=?(以逗号分隔)"
27 a = list(map(int, input(msg).split(','))) # 输入a[i]
28
29 st, ed = st-1, ed-1 # 楼层编号映射到[0, n-1]以便于后续处理
30 if (st == ed):
31     print(0)
32 else:
33     print(bfs(n, st, ed, a))
```