1 Ans

Input Format

The first line of the input contains a single integer T denoting the number of test cases. The description of T test cases follows.

The first and only line of each test case contains six space-separated integers A,B,A1,B1,A2,B2.

Output Format

For each test case, print a single line containing the integer 1 if Chef should switch to the first language, or 2 if Chef should switch to the second language, or 0 if Chef cannot switch to either language.

Constraints

$1 \le T \le 288$

$1 \le A,B,A1,B1,A2,B2$

A,B are distinct

A1,B1,A2,B2 are pairwise distinct

Substaks

Subtask #1(100 points): original constraints

Sample 1:

Input:

3

1 2 2 1 3 4

3 4 2 1 4 3

1 2 1 3 2 4

Output:

1

2

0

1 Ans

Input Format

The first line of the input contains a single integer T denoting the number of test cases. The description of T test cases follows.

The first and only line of each test case contains four space-separated integers A1,A2,A3,A4denoting the difficulty level of four problems.

Output Format

For each test case, print a single line containing one integer – the maximum number of problem sets you can create using the four problems.

Constraints

1≤T≤1000

1≤A1,A2,A3,A4≤10

Subtasks

Subtask #1 (100 points): Original constraints

Sample 1:

Input

3

1 4 3 2

4 5 5 5

2 2 2 2

Output

2

1

0

1 Ans

Simple Python program to compare dates

importing datetime module

Import datetime

```
# date in yyyy/mm/dd format

D1 = datetime.datetime(2018, 5, 3)

D2 = datetime.datetime(2018, 6, 1)


# Comparing the dates will return

# either True or False

Print("d1 is greater than d2 : ", d1 > d2)

Print("d1 is less than d2 : ", d1 < d2)

Print("d1 is not equal to d2 : ", d1 != d2)
```

1 Ans

Class constructors are a fundamental part of object-oriented programming in Python. They allow you to create and properly initialize objects of a given class, making those objects ready to use. Class constructors internally trigger Python's instantiation process, which runs through two main steps: instance creation and instance initialization.

If you want to dive deeper into how Python internally constructs objects and learn how to customize the process, then this tutorial is for you.

In this tutorial, you'll:

Understand Python's internal instantiation process

Customize object initialization using .__init__()

Fine-tune object creation by overriding .__new__()