# 01. Climb the Peaks

You will have to keep information for **all the conquered peaks** if any.

Every day, Alex will **use one portion** of his **daily food supplies** and will **exhaust one** of his **daily stamina**.

First, you will be given **a sequence of numbers, representing the quantities of the daily portions** of food supplies in his backpack.

Afterward, you will be given another **sequence of numbers, representing the quantities of the daily stamina** he will have at his disposal for the next **seven days**.

You have to **sum** the **quantity of the** **last daily food portion** with the **quantity of the first daily stamina**. He will start climbing **from the first** peak in the table below **to the last** one.

* If the **sum is equal to or greater** than the corresponding **Mountain Peak’s Difficulty level from the table below**, it means that the **peak is conquered**. In this case, you should **remove both quantities from the sequences** and **continue** with the **next ones** towardsthe **next peak**.
* If the **sum** is less than the corresponding **Mountain Peak’s Difficulty level** from the table below, the **peak remains unconquered**. You should **remove both quantities from the sequences.** Alex will have to sleep in his tent. On the next day, he will try **the same peak once again**.

|  |  |
| --- | --- |
| **Mountain Peaks** | **Difficulty level** |
| Vihren | 80 |
| Kutelo | 90 |
| Banski Suhodol | 100 |
| Polezhan | 60 |
| Kamenitza | 70 |

Alex will try to conquer as many peaks as he can in seven days. If he manages to climb **all the peaks**, the journey ends and the output is printed on the Console.

Finally, **print** on the Console **all the conquered peaks**(in the order of climbing).

### Input

* On the **first line**, you will receive the **food portions**, **separated** by **a comma and a** **single space (', ')**.
* On the **second line**, you will receive the **stamina**, **separated** by **a comma and a** **single space (', ')**.

### Output

* On the first line – print whether Alex managed to reach his goal and climb all the peaks in his list:
  + If he managed to conquer all: "**Alex did it! He climbed all top five Pirin peaks in one week -> @FIVEinAWEEK**"
  + If he didn't manage to climb all of the peaks: "**Alex failed! He has to organize his journey better next time -> @PIRINWINS**"
* Then, in either case,you need to print **all the conquered peaks** (in the order of climbing) **if any**:

**"Conquered peaks:**

**{peak1}**

* + If there are **no concurred peaks**, do **NOT** print this message.

### Constraints

* All of the given numbers will be valid integers in the range **[0…100]**.

### Examples

|  |  |  |
| --- | --- | --- |
| ****Input**** | ****Output**** | ****Comment**** |
| **40, 40, 40, 40, 40, 40, 40**  **40, 50, 60, 20, 30, 5, 2** | Alex did it! He climbed all top five Pirin peaks in one week -> @FIVEinAWEEK  Conquered peaks:  Vihren  Kutelo  Banski Suhodol  Polezhan  Kamenitza | We start by taking the last daily portion quantity (**40**) and the first stamina quantity (**40**). Their **sum** is **40 + 40 = 80**. After that, we check if the sum is equal to or greater than the **first peak’s difficulty level**. The **sum** of the food portion and the stamina for the day is equal to **the peak’s difficulty level**, so the **peak is conquered**. We **remove both quantities** from the sequences. |

**from** collections **import** deque  
  
food\_portions **=** [int(el) **for** el **in** input().**split**(**', '**)]  
stamina **=** deque(int(el) **for** el **in** input().**split**(**', '**))  
days **=** 7  
peaks\_list **=** deque(  
 [(**'Vihren'**, 80),  
 (**'Kutelo'**, 90),  
 (**'Banski Suhodol'**, 100),  
 (**'Polezhan'**, 60),  
 (**'Kamenitza'**, 70)]  
)  
  
climbed\_peaks **=** []  
**for** \_ **in** range(days)**:  
 if not** peaks\_list**:  
 break** current\_portion **=** food\_portions.**pop**()  
 current\_stamina **=** stamina.**popleft**()  
 current\_peak **=** peaks\_list.**popleft**()  
 current\_capacity **=** current\_portion **+** current\_stamina  
  
 **if** current\_capacity **>=** current\_peak[1]**:** climbed\_peaks.**append**(current\_peak[0])  
 **else:** peaks\_list.**appendleft**(current\_peak)  
  
**if not** peaks\_list**:** print(**"Alex did it! He climbed all top five Pirin peaks in one week -> @FIVEinAWEEK"**)  
**else:** print(**"Alex failed! He has to organize his journey better next time -> @PIRINWINS"**)  
  
**if** climbed\_peaks**:  
 for** peak **in** climbed\_peaks**:** print(peak)