# Python Advanced: Exam Preparation

## Worms and Holes

**[Link to Judge](https://judge.softuni.org/Contests/Practice/Index/4226" \l "0)**

The first line will give you **a sequence of integers representing worms**. Afterwards, you will be given another **sequence of integers representing holes**.

You have to start with the **last worm** and try to match it with the **first hole**.

* If their **values** are **equal**, the worm fits the hole and can get into it. After that, you should **remove both** of them from their sequences. Otherwise, you should **remove** the **hole** and **decrease** the **value** of the **worm**   
  by **3**.
* If the **worm** value becomes **equal to or below** **0**, **remove it** from the sequence **before** trying to **match** it with the hole.

You need to **stop** **matching** when you have **no more worms or holes**.

### Input / Constraints

* On the **first line**, you will receive the integers, representing the **worm size**, **separated** by a **single space**.
* On the **second line,** you will receive the integers, representing the **hole size**, **separated** by a **single space**.
* All given numbers will be valid integers in the range **[1, 50]**.

### Output

* On the **first** line:
* If there are matches print the following:
  + "**Matches: {matchesCount}"**
* If there are no matches print the following:
* **"There are no matches."**
* On the **second** line print:
* If there are no worms left and **all of them fit a hole**:
  + "**Every worm found a suitable hole!**"
* If there are no worms left but only some of them fit a hole:
  + "**Worms left: none**"
* If there are worms left:
  + "**Worms left: {worm1}, {worm2}, (…),{wormn}**"
* On the **third** line print:
* If there are no holes left:
  + "**Holes left: none**"
* If there are holes left:
  + "**Holes left: {hole1}, {hole2}, (…),{holen}**"

### Examples

|  |  |  |
| --- | --- | --- |
| ****Input**** | ****Output**** | ****Comment**** |
| **9 5 8 13**  **13 8 5 6** | **Matches: 3**  **Worms left: 6**  **Holes left: none** | The first pair is the **first hole** with a value of 13 and the **last worm** with a value of 13, their **values are equal**, so the worm gets into the hole **and we remove values** from the **sequences**. Next, there are **two more matches** (8 = 8) and (5 = 5) you should **remove both of them**, too. But the value of the **next worm is 9** and the value of the **next hole is 6**, (9 > 6) so we **reduce** the **worm's value** by 3 and **remove** the **hole**. |
| **17 20 25 25 30**  **9 8 7 21 5 4 3 2 1** | **Matches: 1**  **Worms left: 17, 20, 25, 10**  **Holes left: none** |  |
| **9 8 7 6**  **6 7 8 9** | **Matches: 4**  **Every worm found a suitable hole!**  **Holes left: none** |  |
| **10 10 10 10**  **5** | **There are no matches.**  **Worms left: 10, 10, 10, 7**  **Holes left: none** |  |

## Fishing Competition

**[Link to Judge](https://judge.softuni.org/Contests/Practice/Index/4193" \l "1)**

*You are a longtime captain of an old fishing vessel. The new fishing season begins and you prepare your ship to set sail in search of the big catch…*

You will be given an integer **n** for the **size** of the **fishing area** with a **square** shape. On the next **n** lines, you will receive the **rows** of the **fishing area**. You will be placed in a **random position**, marked with the letter '**S**'. There will be fishing passages on **random positions**, marked with a **single digit**. There will be whirlpools **marked** with '**W**'. All of the empty **positions** will be marked with **'-'**.

Each turn until the "**collect the nets**" command is received you will be given **commands** for **your movement**. Move commands will be: "**up**", "**down**", "**left**", and "**right**".

* If you **move** to a **fish passage**, you **collect** **the amount** **equal** to the **digit** **there**, the **passage disappears** and should be replaced by **'-'**.
* If you **fall into a whirlpool** – the **ship sinks and loses its catch**, the program ends.
* If you **leave the fishing area** (go out of the boundaries of the matrix) depending on the move command you will be **moved to the opposite side of the one you were on**.

/**Example:** In a 3x3 matrix you are at position **[1,2]** and receive the command "**right**" you will be moved to position **[1,0]**./

You need **at least** **20 tons** of fish to be considered a successful season. Keep in mind that even if the **quota is reached** the **ship continues to move**.

### Input

* On the first line, you are given the integer **n** – the size of the **square** matrix.
* The **next n lines** hold the values for every **row**.
* On each of the next lines, you will get a move command.

### Output

* On the first line:
* If the ship **falls into a whirlpool**, print only this message and stop the program:
* **"You fell into a whirlpool! The ship sank and you lost the fish you caught. Last coordinates of the ship: [n,n]"**
* If the **ship reaches** the quota:
* **"Success! You managed to reach the quota!"**
* If the **ship did not reach** the quota:
* **"You didn't catch enough fish and didn't reach the quota!**

**You need {lack of fish} tons of fish more."**

* On the **next** lines.
* If the catch quantity is bigger than zero, print:
* **"Amount of fish caught: {quantity} tons."**

else: **do not print** anything.

* If you **didn't get into a whirlpool**, print the **matrix**.

### Constraints

* The size of the **square** matrix will be between **[2…10].**
* Only the letters '**S**' and '**W**' will be present in the matrix.
* The **fish passages** are represented by **single positive digits** /tons/ between **[1…9]**.
* It is expected that there will only be either **zero** or **one** **whirpool** present, marked with the **letter** - '**W**'.
* Your position will be marked with '**S**'.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 4  ---S  ----  9-5-  34--  down  down  right  down  collect the nets | You didn't catch enough fish and didn't reach the quota! You need 8 tons of fish more.  Amount of fish caught: 12 tons.  ----  ----  --5-  S4-- |
| **Comment** | |
| The first command is **"down"**. The ship moves to position **[1,3]** followed by the command **"down" [2,3]** andthen the command **"right".** The ship leaves the matrix's boundaries and transfers to the opposite side at position **[2,0]**. The ship comes across a **fish passage** with a quantity of 9 tons and gets it. After executing the third command, the **fishing area** will appear as follows:  ----  ----  S-5-  34--  Then you receive the command **"down"** again.You move to the passage of **3** tons and add them to the others **9**. Your catch is **9 + 3 = 12** tons. In the end, you get the command **"collect the nets"** and the program ends. | |

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  S---9  777-1  W333-  11111  -----  down  down  right  down  collect the nets | You fell into a whirlpool! The ship sank and you lost the fish you caught. Last coordinates of the ship: [2,0] |
| **Comment** | |
| The first command is **"down"**. The ship moves to position **[1,0]** and gets **7** tons of fish. Follow the command **"down"** -> **[2,0]** The ship **falls into a whirlpool and sinks.** You lose the entire catch and the program ends. | |

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  S---9  777-1  --5--  11W11  988--  down  down  down  down  down  down  right  right  right  collect the nets | Success! You managed to reach the quota!  Amount of fish caught: 31 tons.  ----9  ---S1  --5--  -1W11  -88-- |
| **Comment** | |
| Result is: **7 + 1 + 9 +7 + 7 = 31**. You succeeded! | |

## Enrollment

**[Link to Judge](https://judge.softuni.org/Contests/Practice/Index/4081" \l "2)**

*You are planning to study at Software University.   
You need to choose classes so that you gather enough credits to graduate successfully.*

Write a function called **gather\_credits** that **receives information** about credits needed, courses, and their credits, and **returns the result**. The function will receive a **different number of arguments**. The arguments will be passed as follows:

* The first argument will be the **number of credits** you need - an **integer** in the **range [0, 200];**
* The following arguments will be the **tuples with two elements** - the **first** one is the **course name (string)**, and the **second** one is the **course** **credits** **(integer)**;

After receiving the information and calling the function, the program should **start tracking the enrollment process**:

* Take the **course's name** from each tuple **successively** and **if you need more credits**, **enroll in it**, and proceed to the next one.
* If a course has **already been enrolled in**, **ignore** it, and proceed to the next one.
* If you have **reached the needed number of credits**, **STOP enrolling**!

In the end:

* If you've managed to **gather the needed credits**, return the message, including the enrolled courses on a new line:

**"Enrollment finished! Maximum credits: {gathered\_credits}.**

**Courses: {course1, course2, …, courseN}"**

* + return the courses' names **sorted alphabetically, in ascending** order.
* **Otherwise**, return the message:

**"You need to enroll in more courses! You have to gather {credits\_shortage} credits more."**

***Note: Submit only the function in the judge system***

### Input

* There will be **no input from the console**, just parameters passed to your function.

### Output

* Return one of the **strings** **shown above** depending on the result.

### Constraints

* The **first** argument will always be an **integer**.
* Each **tuple** given will always contain the **course name** with its **credits**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| print(gather\_credits(  80,  ("Basics", 27),  )) | You need to enroll in more courses! You have to gather 53 credits more. |
| print(gather\_credits(  80,  ("Advanced", 30),  ("Basics", 27),  ("Fundamentals", 27),  )) | Enrollment finished! Maximum credits: 84.  Courses: Advanced, Basics, Fundamentals |
| print(gather\_credits(  60,  ("Basics", 27),  ("Fundamentals", 27),  ("Advanced", 30),  ("Web", 30)  )) | Enrollment finished! Maximum credits: 84.  Courses: Advanced, Basics, Fundamentals |