## Problem 1. Sweet Dessert

Ivancho and his girlfriend are **throwing a party**. She plans **to cook her favorite dessert**. She asks Ivancho to **buy** the **needed products**. The **number of desserts** depends on **how many people will be coming**. She can prepare the dessert **in portions of 6**. If there are **5 guests** coming, she will still **cook 6 portions,** **10 guests – cook 12**. The products for the dessert are **bananas, eggs and berries**. For **a set of 6** she needs **2 bananas, 4 eggs and 0.2 kilos berries**.

You will be given **the amount of money Ivancho has**, the **number of guests** and the **prices of the products**. You have to help Ivancho **calculate** if the **cash** he has is **enough to buy all of the products**, or how much more money he needs.

### Input

The input data should be read from the console. It will consist of **exactly 5 lines**:

* The **amount of cash** Ivancho has – **floating-point number** in **range [0.00…1,000,000,000.00]**
* The **number of guests – integer in range [0…1,000,000,000]**
* The **price of bananas** for a **single unit – floating-point number** in **range [0.00…1,000.00]**
* The **price of eggs** for a **single unit – floating-point number** in **range [0.00…1,000.00]**
* The **price of berries** for a **kilo – floating-point number** in **range [0.00…1,000.00]**

The **input data will always be valid** and in the format described. **There is no need to check it explicitly**.

### Output

The output should be printed on the console.

* **If the calculated price of the products is less or equal to the money Ivancho has:**
  + “Ivancho has enough money - it would cost {the cost of the products}lv.”
* **If the calculated price of the products is more than the money Ivancho has:**
  + “Ivancho will have to withdraw money - he will need {neededMoney}lv more.”
* **All prices** must be **rounded to two digits after the decimal point.**

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 10  12  0.35  0.20  4.50 | Ivancho has enough money - it would cost 4.80lv. | For 12 guests – 2 sets of 6 portions  Needed product:  2\*(2 bananas), 2\*(4 eggs), 2\*(0.2 kilos berries)  2\*(2\*0.35) + 2\*(4\*0.20) + 2\*(0.2\*4.50) = 4.80  4.80 <= 10 – the money will be enough. |
| **Input** | **Output** | **Comments** |
| 20  33  0.60  0.50  10 | Ivancho will have to withdraw money - he will need 11.20lv more. | For 33 guests – 6 sets of 6 portions  Needed product:  6\*(2 bananas), 6\*(4 eggs), 6\*(0.2 kilos berries)  6\*(2\*0.60) + 6\*(4\*0.50) + 6\*(0.2\*10.00) = 31.20  31.20 > 20 – need 11.20 lv. more. |

## Problem 2. Game of Names

Write a program to **calculate points** **for all players** and **find who the winner is**. You will be given **the count of the players**, **their names** and **initial scores**. **Score** for every player **depends on his name**. To the **player score** **add** or **subtract** the **ASCII code** **of** **each letter**. If **ASCII code** is **even** **add** it to the score. If is **odd** – **subtract** it from the score. Find **the one with highest score** and print his name and score on the console. If **two or more players** are with **same points** – **the winner is the first one**.

### Input

On the **first input line** you will be given **number N -** **the count of players**.

On the **next 2\*N lines** you will be given player name and hi initial score.

### Output

**The output** should be printed on the console and consist **the name of the winner** and **his score** in the following format:

“The winner is {name} - {points} points”

### Constraints

* **N – the count of players** will be a **positive integer** in the range **[1...100]**
* **Names** will be **strings** with **length between 3 and 30**
* **The score** **for each player** will be a **integer** in the range **[-100,000...100,000]**

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 3  Bojidar  123  Preslav  123  Pesho  123 | The winner is Preslav - 230 points | B(66)o(111)j(106)i(105)d(100)a(97)r(114) Initial points 123 scores  123 +66 -111 +106 -105 +100 -97 +114 = 196  P(80)r(114)e(101)s(115)l(108)a(97)v(118)  Initial points 123 scores  123 +80 +114 -101 -115 +108 -97 +118 = 230  P(80)e(101)s(115)h(104)o(111)  Initial points 123 scores  123 +80 -101 -115 +104 -111 = -20  Preslav(230) > Bojidar(196) > Pesho(-20) |

## Problem 3. Strawberry

Your task is to write a program, which **reads a number N** from the console and **draws a strawberry**, which **size depends on N**. See the examples below to understand your task better.

### Input

On the **only input line** you will be given the **positive** **number N** – **odd number** in the range **[1 … 17]**

### Output

The output should be printed on the console. Use the **“#” (number sign)** for the **outlines of the strawberry**, **“.”** for the **inside of the strawberry**. For **the leaves:** **left** ones – **“\”**, **middle** ones – **“|”** and **right** ones – **“/”**

### Examples

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Input** | **Output** | **Input** | **Output** | **Input** | **Output** |
| 5 | \-----|-----/  --\---|---/--  -----#.#-----  ---#.....#---  -#.........#-  #...........#  -#.........#-  --#.......#--  ---#.....#---  ----#...#----  -----#.#----- | 7 | \-------|-------/  --\-----|-----/--  ----\---|---/----  -------#.#-------  -----#.....#-----  ---#.........#---  -#.............#-  #...............#  -#.............#-  --#...........#--  ---#.........#---  ----#.......#----  -----#.....#-----  ------#...#------  -------#.#------- | 9 | \---------|---------/  --\-------|-------/--  ----\-----|-----/----  ------\---|---/------  ---------#.#---------  -------#.....#-------  -----#.........#-----  ---#.............#---  -#.................#-  #...................#  -#.................#-  --#...............#--  ---#.............#---  ----#...........#----  -----#.........#-----  ------#.......#------  -------#.....#-------  --------#...#--------  ---------#.#--------- |

## Problem 4. Array Modifier

You are given **an array with integers**. Write a program to **modify the elements** after **receive the commands** “**swap**”, “**multiply**” or “**decrease**”.

* “swap {index1} {index2}” take **two elements** and **swap their places**.
* “multiply {index1} {index2}” take **element at the 1st index** and **multiply** **it** **with element at 2nd index**. **Save the product at the 1st index.**
* “decrease” **decreases** **all elements** in the array **with 1**.

### Input

On the **first input line** you will be given **the initial array values** separated by a single space.

On the **next lines** you will receive commands **until** you receive the **command “end”**. The **commands are** as follow:

* “swap {index1} {index2}”
* “multiply {index1} {index2}”
* “decrease”

### Output

**The output** should be printed on the console and consist **element** **of the** **modified array** – **separated by “, “**(**comma and single space**).

### Constraints

* Commands will be: “**swap**”, “**multiply**”, “**decrease**” and “**end**”
* **Elements of the array** will be **integer numbers** in the range **[-231...231]**
* **Count of the array elements** will be in the range **[2...100]**
* **Indexes** **will be always in the range of the array**

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 23 -2 321 87 42 90 -123  swap 1 3  swap 3 6  swap 1 0  multiply 1 2  multiply 2 1  decrease  end | 86, 7382, 2369942, -124, 41, 89, -3 | 23 -2 321 87 42 90 -123 – initial values  swap 1(-2) and 3(87) ▼  23 87 321 -2 42 90 -123  swap 3(-2) and 6(-123) ▼  23 87 321 -123 42 90 -2  swap 1(87) and 0(23) ▼  87 23 321 -123 42 90 -2  multiply 1(23) 2(321) = 7383 ▼  87 7383 321 -123 42 290 -2  multiply 2(321) 1(7383) = 2369943 ▼  87 7383 2369943 -123 42 90 -2  decrease – all - 1 ▼  86 7383 2369942 -124 41 89 -3 |

## Problem 5. Multiply Targeted Cell

Write a program which reads from the console **dimensions of a matrix** and **matrix elements values**. Get a **targeted cell** and **multiply** **its value** with **the sum of all neighboring cells**. The **neighboring cells** must **change their values too**. **Each one** should be **the product** of **its initial value** and **the initial value of the targeted cell**. Then **print on the console updated matrix**.

### Input

The input data should be read from the console:

* The **first line** holds the number of **rows – R** and **columns – C,** separated by space.
* The **next R lines** hold the **matrix values**. Each line holds **C** **integers**, separated by space.
* The **last line** holds **the position** (targeted **row** and targeted **col**) **of the targeted cell**, separated by space

The **input data will always be valid** and in the format described. **There is no need to check it explicitly**.

### Output

The output should be printed on the console. The elements of each row should be separated by space.

### Constraints

* The **dimensions** of the matrix **(R and C)** will be a **positive integer numbers** in the range **[3...20]**.
* The **values of the cells** will be an **integer numbers** in range **[-16,300... 16,300]**.
* The **targeted row** will be an **integer number** in the range **[1...R-2]**.
* The **targeted column** will be an **integer number** in the range **[1...C-2]**.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 5 5  10 12 14 16 17  10 12 14 16 17  10 12 14 16 17  10 12 14 16 17  10 12 14 16 17  2 2 | 10 12 14 16 17  10 168 196 224 17  10 168 1568 224 17  10 168 196 224 17  10 12 14 16 17 | Targeted cell is [2,2] = 14  The sum all neighboring cells is:  12 + 14 + 16 + 12 + 16 + 12 + 14 + 16 = 112  The targeted cell new value = 14 \* 112 = 1568  Neighboring cells new values:  [1,1]=12\*14=168; [1,2]=14\*14=196; [1,3]=16\*14=224;  [2,1]=12\*14=168; [2,3]=14\*14=224;  [3,1]=12\*14=168; [3,2]=14\*14=196; [3,3]=16\*14=224 |
| **Input** | **Output** | |
| 6 4  0 105 420 480  1 143 624 744  2 182 628 488  3 226 326 538  4 263 376 406  5 -1 -2 -3  4 2 | 0 105 420 480  1 143 624 744  2 182 628 488  3 84976 122576 202288  4 98888 659128 152656  5 -376 -752 -1128 | |