Medium Questions

1.TR**AINING LAB**: A training lab has a rectangular size l x w meters, without columns on the inside. The hall is divided into two parts- left and right, with a hallway approximately in the middle. In both parts, there are **rows with desks**. In the back of the hall, there is a big **entrance door**. In the front, there is a **podium** for the lecturer. A single **working place** takes up **70 x 120 cm** (a table with size 70 x 40 cm + space for a chair with size 70 x 80 cm). **The hallway** width is at least **100 cm**. It is calculated that due to the **entrance door** (which has 160 cm opening), **exactly one working space is lost**, and due to the **podium** (which has size of 160 x 120 cm), exactly **two working spaces** are lost. Write a program that reads the size of the training lab as input parameters and calculates the **number of working places in it** (look at the figure).

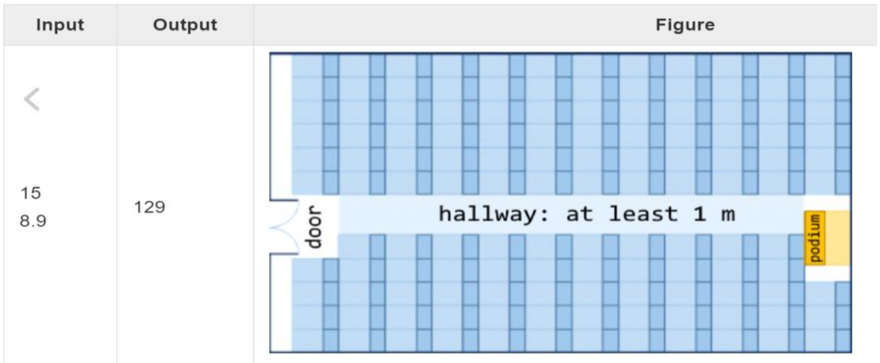
Input Data

Two numbers are read from the console, one per line: l (length in meters) and w (width in meters). Constraints: **3 ≤ w ≤ l ≤ 100.**

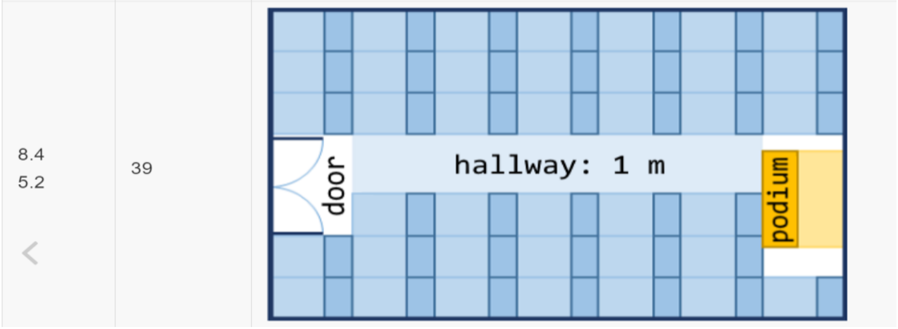
Output Data

Print an integer: the **number of working places** in the training lab.

Sample Input and Output



Input	Output	Figure
15 8.9	129	



8.4 5.2	39	

2. Vegetable Market: A gardener is selling his harvest on the vegetables market. He is selling **vegetables for N coins per kilogram** and **fruits for M coins per kilogram**. Write a program that **calculates the earnings of the harvest in Euro** (EUR). Assume the EUR / coin rate is fixed: **1 Euro** == **1.94 coins**.

Input Data

Four numbers are read from the console, one per line:

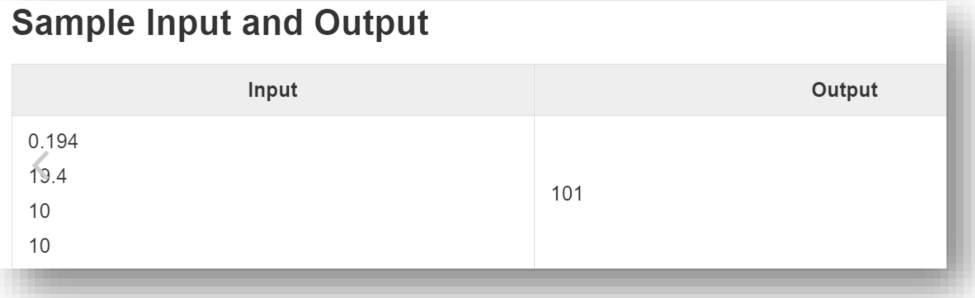
- First line: **vegetable price** per kilogram – a floating-point number.
- Second line: **fruit price** per kilogram – a floating-point number.

- Third line: total **kilograms of vegetables** – an integer.
- Fourth line: total **kilograms of fruits** – an integer.

Constraints: all numbers will be within the range from 0.00 to 1000.00.

Output Data

Print on the console **one floating-point number: the earnings of all fruits and vegetables in Euro**.



Sample Input and Output

Input	Output
0.194 19.4 10 10	101

3.Change Tiles: The tiles on the ground in front of an apartment building **need changing**. The ground has a **square shape with side of N meters**. The tiles are **"W" meters wide** and **"L" meters long**. There is one bench on the ground with **width of "M" meters and length of "O" meters**. The tiles under it do not need to be replaced. Each tile is replaced for **0.2 minutes**.

Write a program that **reads the size of the ground, the tiles and the bench from the console**, and calculates how many tiles are needed to cover the ground and what is **the total time for replacing the tiles**.

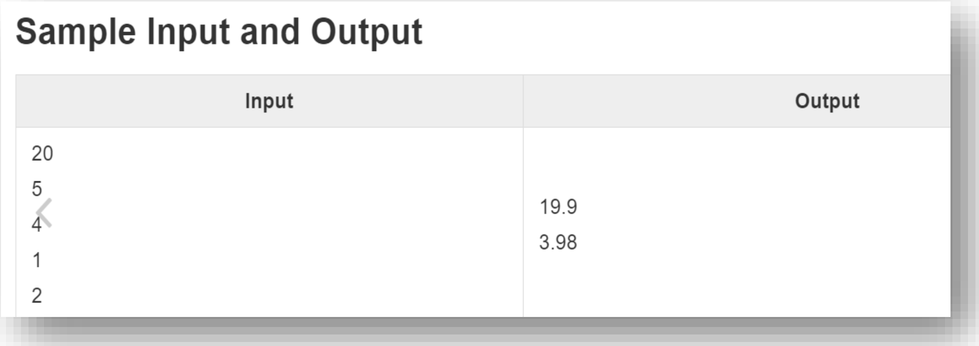
Input Data

The input data comes as **5 numbers**, which are read from the console:

- **N – length** of **a side** of **the ground** within the range of [**1 … 100**].
- **W – width** per **tile** within the range of [**0.1 … 10.00**].
- **L – length** per **tile** within the range of [**0.1 … 10.00**].
- **M – width** of **the bench** within the range of [**0 … 10**].
- **O – length** of **the bench** within the range of [**0 … 10**].

Output Data

Print on the console **two numbers**: **number of tiles needed** for the repair and the **total time for changing them**, each on a new line.



Sample Input and Output

Input	Output
20 5 4 1 2	19.9 3.98

4. Money: Peter lives in Bulgaria and is keen about money exchange, trading, cryptocurrencies and financial markets. Some time ago, Peter **bought Bitcoins** (BTC) and paid for them in **Bulgarian levs** (BGN). Now, he is going on vacation in Europe and **he needs Euro** (EUR). Apart from the Bitcoins, he has **Chinese yuans** (CNY) as well. Peter wants to **exchange his money for Euro** for the tour. Write a program that calculates **how much Euro he can buy**, depending on the following **exchange rates**:

- **1 Bitcoin (BTC) = 1168 BGN**
- **1 Chinese yuan (CNY) = 0.15 dollars (USD)**
- **1 dollar (USD) = 1.76 BGN**
- **1 Euro (EUR) = 1.95 BGN**

The exchange office has **commission fee within 0% to 5%** from the final sum in Euro.

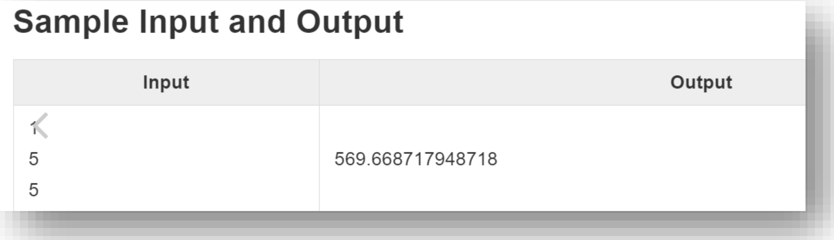
Input Data

Three numbers are read from the console:

- On the first line – **number of Bitcoins**. Integer within the range of [**0 … 20**].
- On the second line – **number of Chinese yuans**. Floating-point number within the range of [**0.00 … 50 000.00**].
- On the third line – **commission fee**. Floating-point number within the range of [**0.00 … 5.00**].

Output Data

Print one number on the console – **the result of the exchange of currencies**. Rounding is not necessary.



Sample Input and Output

Input	Output
1 5 5	569.668717948718

5.Daily Earnings: Ivan is a programmer in an **American company**, and he **works** at home **approximately N days per month** by earning **approximately M dollars per day**. At the end of the year, Ivan **gets a bonus**, which **equals 2.5 of his monthly salaries**. In addition, **25% of his annual salary goes for taxes**.

Write a program that calculates **what is the amount of Ivan's net average earnings in EUR per day**, as he spends them in Europe. It is accepted that **one year has exactly 365 days**. The exchange rate of US **dollar** (USD) to **Euro** (EUR) will be **read from the console**.

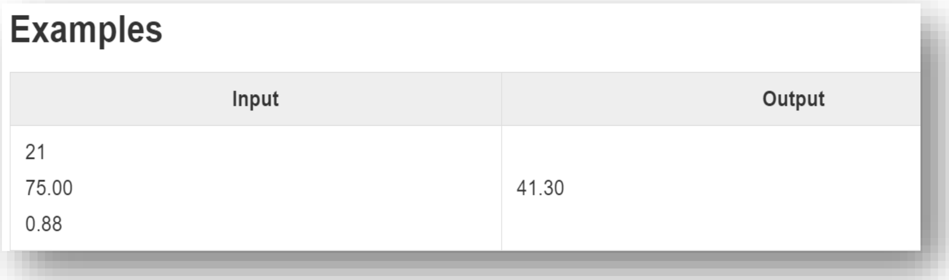
Input Data

Three numbers are read from the console.

- On the first line – **work days per month**. An integer within the range of [**5 … 30**].
- On the second line – **daily earnings**. A floating-point number within the range of [**10.00 … 2000.00**].
- On the third line – **exchange rate of USD to EUR** /1 dollar = X euro/. A floating-point number within the range of [**0.05 … 4.99**].

Output Data

Print **one number** on the console – **the daily earnings in EUR**. The result will be **rounded up to the second digit after the decimal point**.



Examples

Input	Output
21 75.00 0.88	41.30

6.Transportation Price

A student has to travel **n kilometers**. He can choose between **three types of transportation**:

- **Taxi**. Starting fee: **0.70** EUR. Day rate: **0.79** EUR/km. Night rate: **0.90** EUR/km.
- **Bus**. Day / Night rate: **0.09** EUR/km. Can be used for distances of minimum **20** km.
- **Train**. Day / Night rate: **0.06** EUR/km. Can be used for distances of minimum **100** km.

Write a Program that reads the number of **kilometers n** and **period of the day** (day or night) and calculates **the price for the cheapest transport**.

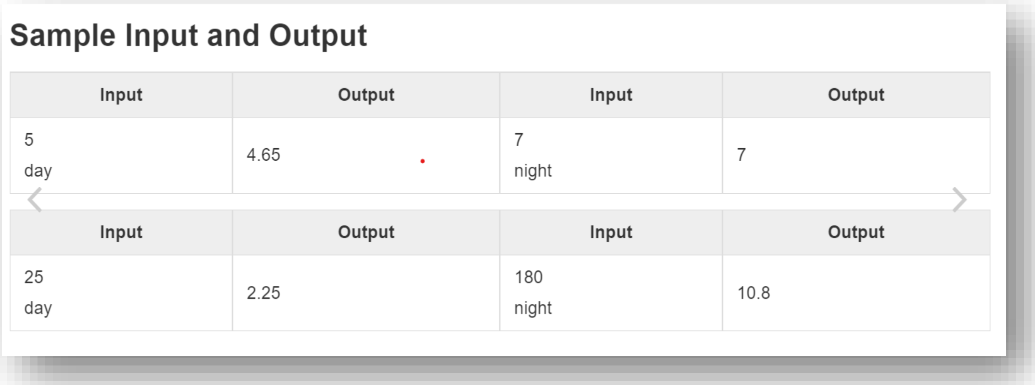
Input Data

Two lines are read from the console:

- The first line contains a number **n** – number of kilometers – an integer in the range of [**1 … 5000**].
- The second line contains the word “**day**” or “**night**” – traveling during the day or during the night.

Output Data

Print on the console **the lowest price** for the given number of kilometers.



Sample Input and Output

Input	Output	Input	Output
5 day	4.65	7 night	7

Input	Output	Input	Output
25 day	2.25	180 night	10.8

7.Problem: Pipes in Pool

A pool with volume V fills up via two pipes. Each pipe has a certain flow rate (the liters of water, flowing through a pipe for an hour). A worker starts the pipes simultaneously and goes out for N hours. Write a program that finds the state of the pool the moment the worker comes back.

Input Data

Four lines are read from the console:

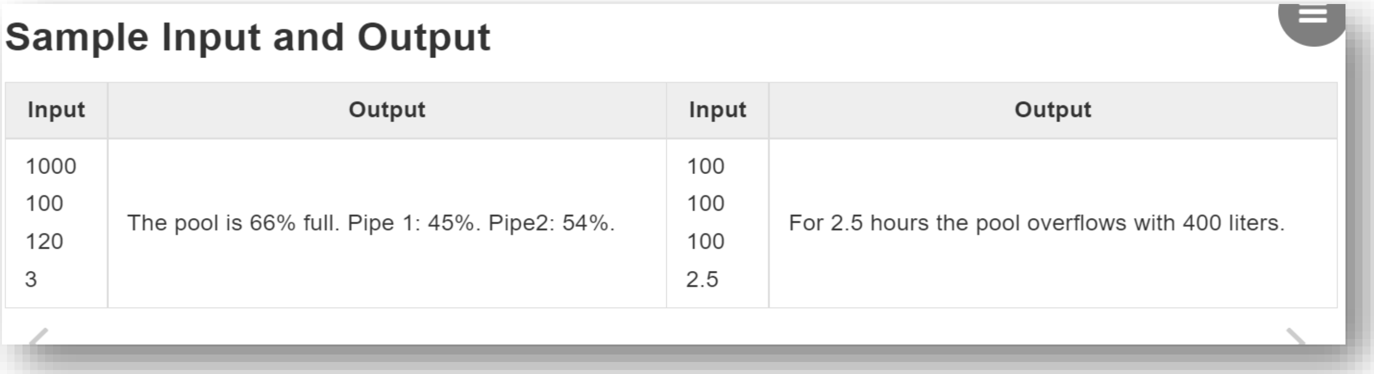
- The first line contains a number **V – the volume of the pool in liters** – an integer in the range of [**1 … 10000**].
- The second line contains a number **P1 – the flow rate of the first pipe per hour** – an integer in the range of [**1 … 5000**].
- The third line contains a number **P2 – the flow rate of the second pipe per hour** – an integer in the range of [**1 … 5000**].
- The fourth line contains a number **H – the hours that the worker is absent** – a floating-point number in the range of [**1.0 … 24.00**].

Output Data

Print on the console **one of the two possible states**:

- To what extent the pool has filled up and how many percent each pipe has contributed with. All percent values must be formatted to an integer (without rounding).
  - "The pool is **[x]**% full. Pipe 1: **[y]**%. Pipe 2: **[z]**%."
- If the pool has overflown – with how many liters it has overflown for the given time – a floating-point number.
  - "For **[x]** hours the pool overflows with **[y]** liters."

Have in mind that due to **the rounding to an integer**, there is **data loss** and it is normal **the sum of the percents to be 99%, not 100%**.



Sample Input and Output

Input	Output	Input	Output
1000 100 120 3	The pool is 66% full. Pipe 1: 45%. Pipe2: 54%.	100 100 100 2.5	For 2.5 hours the pool overflows with 400 liters.

8.Problem: Sleepy Tom Cat

Tom Cat likes to sleep all day but, unfortunately, his owner is always playing with him whenever he has free time. In order to sleep well, **the norm of games** that Tom has is **30 000 minutes per year**. The time for games he has **depends on the holidays that his owner has**:

- During **work days**, his owner plays with him **63 minutes per day**.
- During **holidays**, his owner plays with him **127 minutes per day**.

Write a program that reads **the number of holidays** and prints whether **Tom can sleep well** and how much **the difference from the norm** for the current year is. It is assumed that **there are 365 days in one year**.

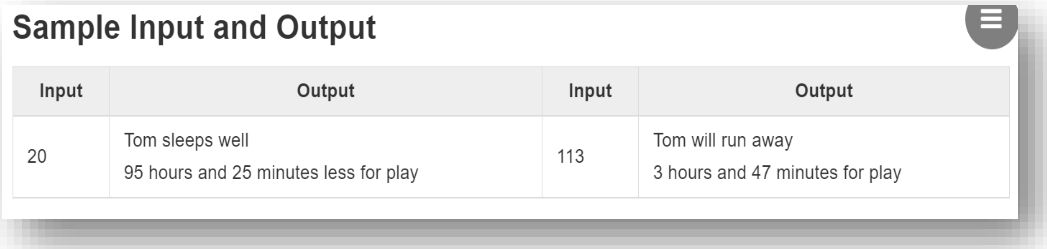
Input Data

The input is read from the console and consists of an integer – **the number of holidays** in the range of [**0 … 365**].

Output Data

Two lines have to be printed on the console:

- If Tom's time for games **is above the norm** for the current year:
  - **On the first line** print: **“Tom will run away”**
  - **On the second line** print the difference from the norm in the format: **“{H} hours and {M} minutes more for play”**
- If the time for games of Tom **is below the norm** for the current year:
  - **On the first line** print: **“Tom sleeps well”**
  - **On the second line** print the difference from the norm in the format: **“{H} hours and {M} minutes less for play”**



Sample Input and Output

Input	Output	Input	Output
20	Tom sleeps well 95 hours and 25 minutes less for play	113	Tom will run away 3 hours and 47 minutes for play

9. Problem: Harvest

In a vineyard with area X square meters, 40% of the harvest goes for wine production. Y kilograms of grapes are extracted from 1 m^2 vineyard. 2,5 kg of grapes are needed for 1 liter of wine. The wanted quantity of wine for sale is Z liters.

Write a program that calculates how much wine can be produced and whether that quantity is enough. If it is enough, the rest is divided between the vineyard workers equally.

Input Data

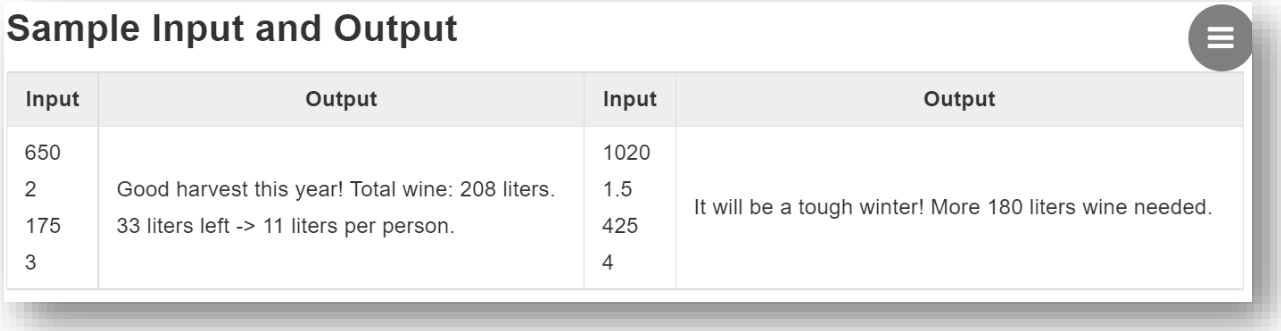
The input data is read from the console and consists of **exactly 4 lines**:

- First line: **X** m^2 **– the vineyard size** – an integer in the range of [**10 … 5000**].
- Second line: **Y grapes for one** m^2 – an integer in the range of [**0.00 … 10.00**].
- Third line: **Z needed liters of wine** – an integer in the range of [**10 … 600**].
- Fourth line: **number of workers** – an integer in the range of [**1 … 20**].

Output Data

The following has to be printed on the console:

- If the **produced** wine is **less than the needed quantity**:
    - **"It will be a tough winter! More {insufficient wine} liters wine needed.**"
      * **The result** has to be **rounded down to the nearest integer**.
- If **the produced** wine is **more than the needed quantity**:
    - **"Good harvest this year! Total wine: {total wine} liters."**
      * **The result** has to be **rounded down to the nearest integer**.
    - **"{Wine left} liters left -> {wine for one worker} liters per person."**
      * **Both of the results** have to be **rounded up to the higher integer**.



Sample Input and Output

Input	Output	Input	Output
650 2 175 3	Good harvest this year! Total wine: 208 liters. 33 liters left -> 11 liters per person.	1020 1.5 425 4	It will be a tough winter! More 180 liters wine needed.

10.Program: Firm

A firm **gets a request for creating a project for which** a certain number of hours **are needed**. The firm has **a certain number of days**. **During 10% of the days, the workers** are being trained and **cannot work** on the project. A normal **working day is 8 hours long**. The project is important for the firm and every worker must work on it with **overtime of 2 hours per day**.

The hours must be **rounded down to the nearest integer** (for example, **6.98 hours** are rounded to **6 hours**).

Write a program that calculates whether **the firm can finish the project on time** and **how many hours more are needed or left**.

Input Data

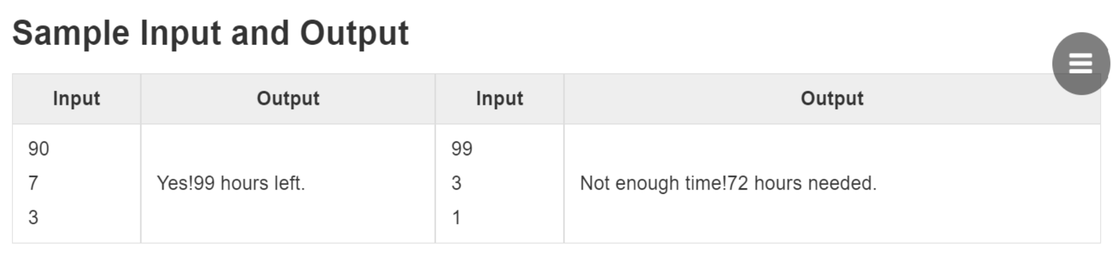
The input data is read from **the console** and contains **exactly three lines**:

- On **the first** line are **the needed hours** – **an integer in the range of** [**0 … 200 000**].
- On **the second** line are **the days that the firm has** – **an integer in the range of** [**0 … 20 000**].
- On **the third** line are **the number of all workers** – **an integer in the range of** [**0 … 200**].

Output Data

Print **one line** on **the console**:

- If **the time is enough**:
  - **"Yes!{the hours left} hours left."**.
- If **the time is NOT enough**:
  - **"Not enough time!{additional hours} hours needed."**.



Sample Input and Output

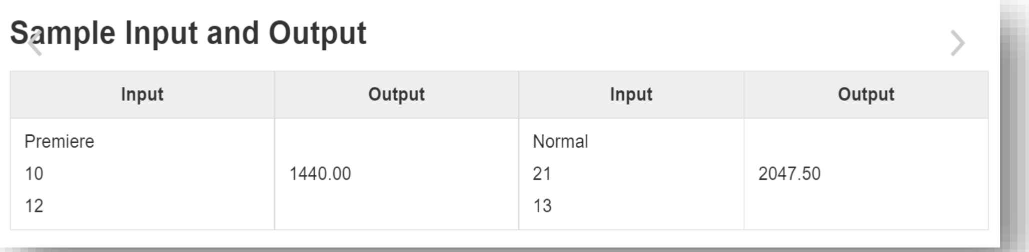
Input	Output	Input	Output
90 7 3	Yes!99 hours left.	99 3 1	Not enough time!72 hours needed.

11.Cinema

In a cinema hall the chairs are ordered in a **rectangle** shape in **r** rows and **c** columns. There are three types of screenings with tickets of **different** prices:

- **Premiere** – a premiere screening, with price **12.00** EUR.
- **Normal** – a standard screening, with price **7.50** EUR.
- **Discount** – a screening for children and students on a reduced price – **5.00** EUR.

Write a program that enters a **type of screening** (string), number of **rows** and number of **columns** in the hall (integer numbers) and calculates **the total income** from tickets from a **full hall**. The result has to be printed in the same format as in the examples below – rounded up to 2 digits after the decimal point.



Sample Input and Output

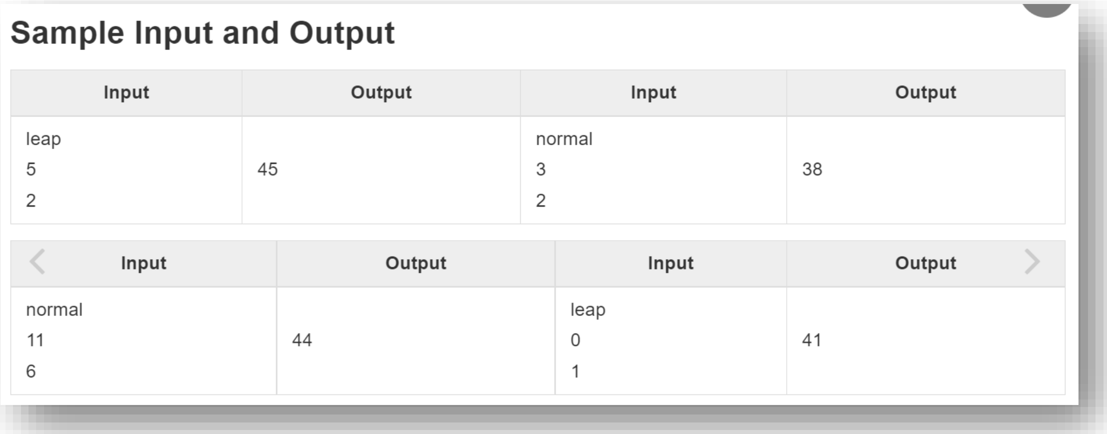
Input	Output	Input	Output
Premiere 10 12	1440.00	Normal 21 13	2047.50

12 .Volleyball

Vladimir is a student, lives in Sofia and goes to his hometown from time to time. He is very keen on volleyball, but is busy during weekdays and plays **volleyball** only during **weekends** and on **holidays**. Vladimir plays **in Sofia** every **Saturday**, when **he is not working**, and **he is not traveling to his hometown** and also during **2/3 of the holidays**. He travels to his **hometown h times** a year, where he plays volleyball with his old friends on **Sunday**. Vladimir **is not working 3/4 of the weekends**, during which he is in Sofia. Furthermore, during **leap years** Vladimir plays **15% more** volleyball than usual. We accept that the year has exactly **48 weekends**, suitable for volleyball. Write a program that calculates **how many times Vladimir has played volleyball** through the year. **Round the result** down to the nearest whole number (e.g. 2.15 -> 2; 9.95 -> 9).

The input data is read from the console:

- The first line contains the word “**leap**” (leap year) or “**normal**” (a normal year with 365 days).
- The second line contains the integer **p** – the count of holidays in the year (which are not Saturday or Sunday).
- The third line contains the integer **h** – the count of weekends, in which Vladimir travels to his hometown.



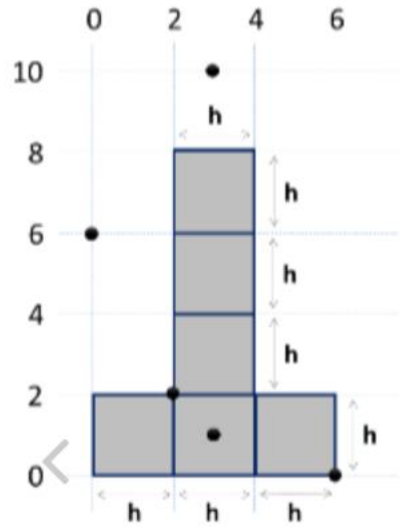
Sample Input and Output

Input	Output	Input	Output
leap 5 2	45	normal 3 2	38

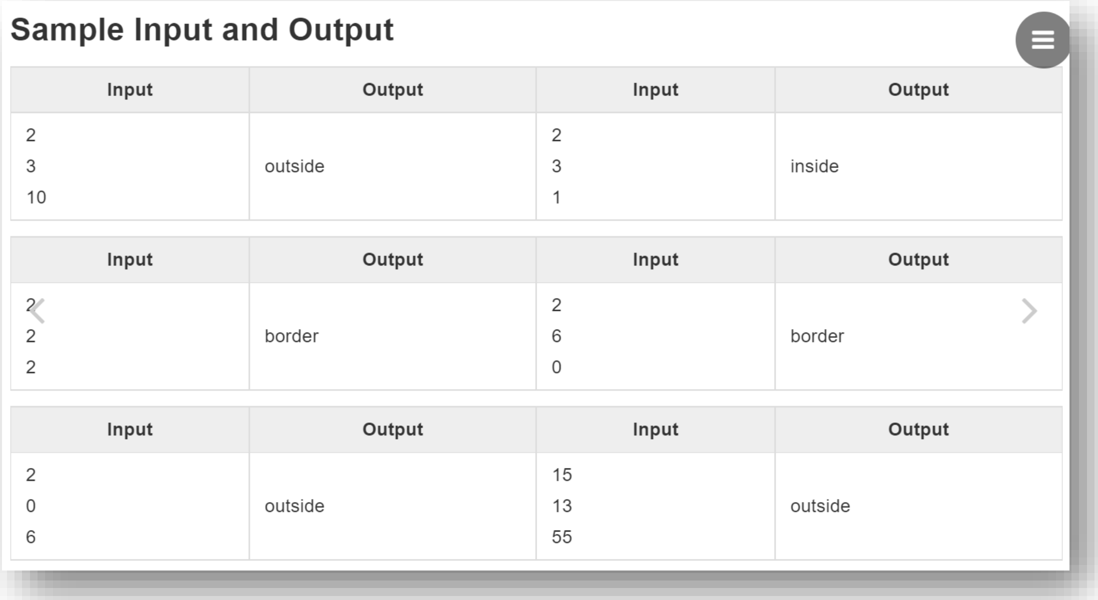
Input	Output	Input	Output
normal 11 6	44	leap 0 1	41

13: Point in the Figure

The figure consists of **6 blocks with size h * h**, placed as in the figure below. The lower left angle of the building is on position {0, 0}. The upper right angle of the figure is on position {**2*h**, **4*h**}. The coordinates given in the figure are for **h = 2**:



Write a program that enters an integer **h** and the coordinates of a given **point {x, y}** (integers) and prints whether the point is inside the figure (**inside**), outside of the figure (**outside**) or on any of the borders of the figure (**border**).

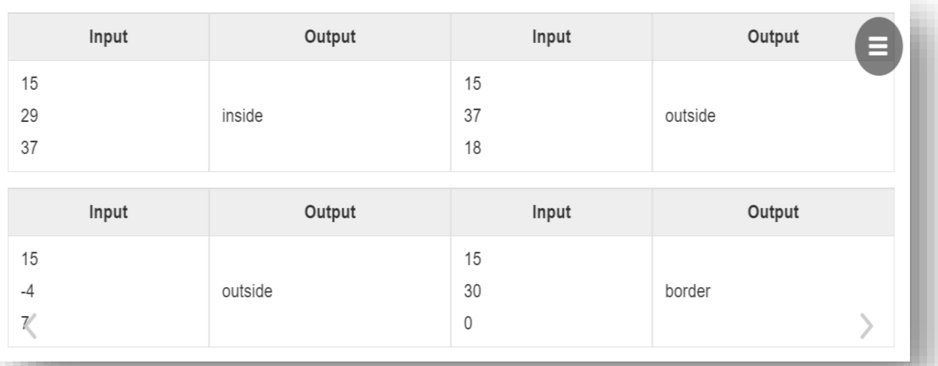


Sample Input and Output

Input	Output	Input	Output
2 3 10	outside	2 3 1	inside

Input	Output	Input	Output
2 2 2	border	2 6 0	border

Input	Output	Input	Output
2 0 6	outside	15 13 55	outside



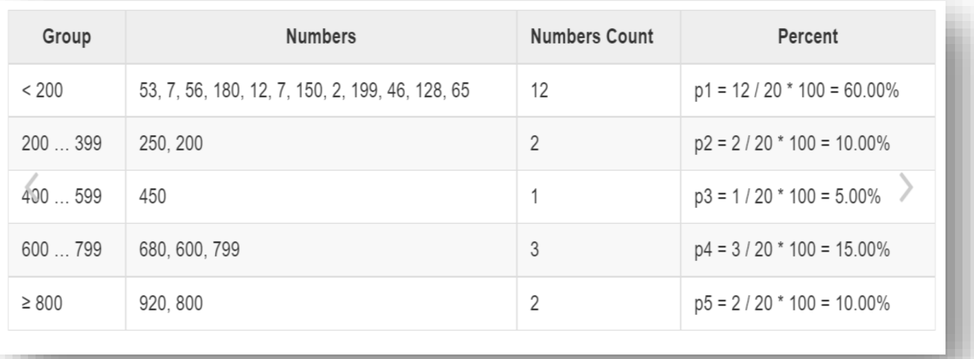
Input	Output	Input	Output
15 29 37	inside	15 37 18	outside

Input	Output	Input	Output
15 -4 7	outside	15 30 0	border

14.Histogram

We have **n integer numbers** within the range of [**1 … 1000**]. Some percent of them **p1** are under 200, another percent **p2** are from 200 to 399, percent **p3** are from 400 to 599, percent **p4** are from 600 to 799 and the rest **p5** percent are from 800 upwards. Write a program that calculates and prints the percentages **p1**, **p2**, **p3**, **p4** and **p5**.

Example: we have n = **20** numbers: 53, 7, 56, 180, 450, 920, 12, 7, 150, 250, 680, 2, 600, 200, 800, 799, 199, 46, 128, 65. We get the following distribution and visualization:



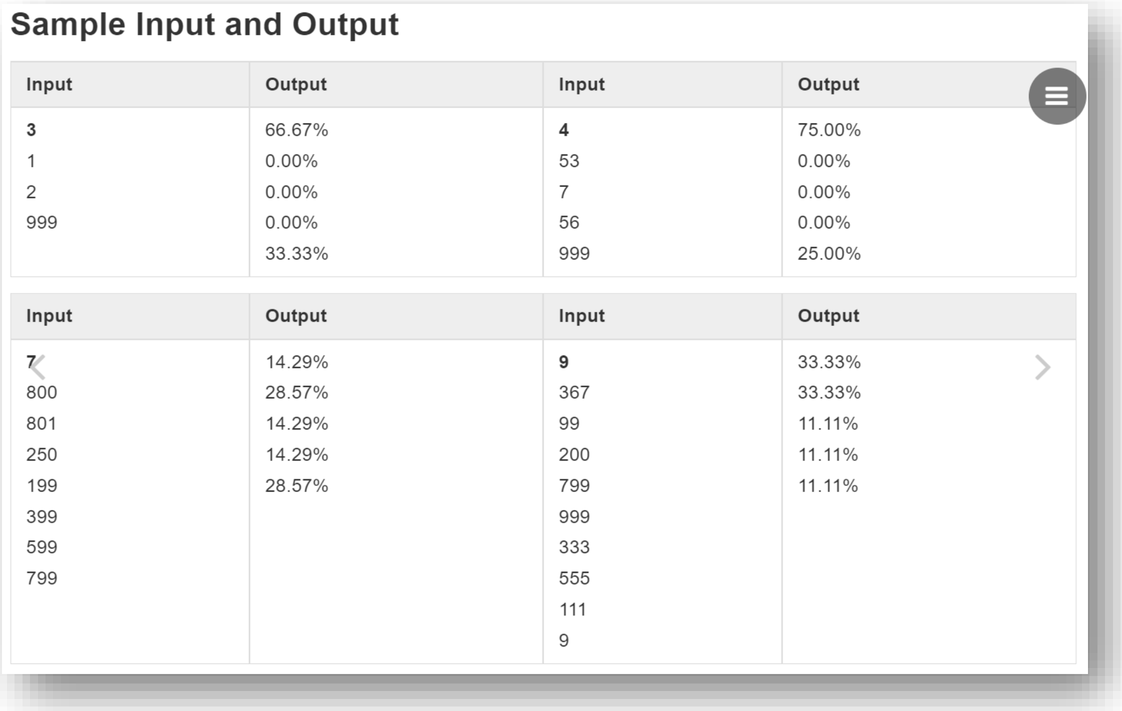
Group	Numbers	Numbers Count	Percent
< 200	53, 7, 56, 180, 12, 7, 150, 2, 199, 46, 128, 65	12	p1 = 12 / 20 * 100 = 60.00%
200 … 399	250, 200	2	p2 = 2 / 20 * 100 = 10.00%
400 … 599	450	1	p3 = 1 / 20 * 100 = 5.00%
600 … 799	680, 600, 799	3	p4 = 3 / 20 * 100 = 15.00%
≥ 800	920, 800	2	p5 = 2 / 20 * 100 = 10.00%

Input Data

On the first line of the input there is an integer **n** (1 ≤ **n** ≤ 1000) that represents the count of lines of numbers that will be passed. On each of the following **n lines** we have **one integer** within range of [**1 … 1000**] – numbers, on which we have to calculate the histogram.

Output Data

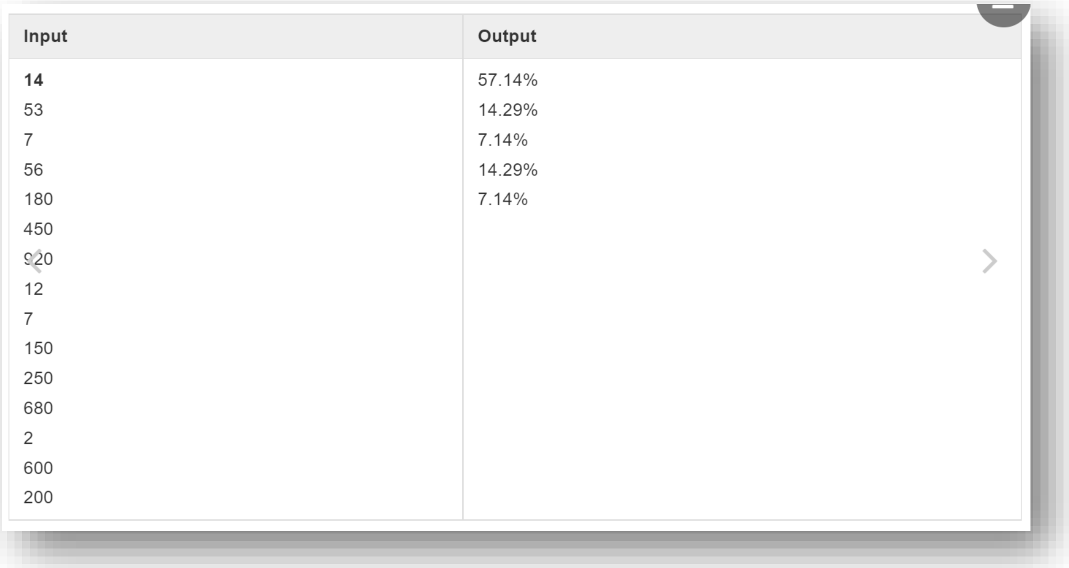
Print on the console a **histogram that consists of 5 lines**, each of them containing a number within the range of [0% … 100%], formatted up to two digits after the decimal point (for example 25.00%, 66.67%, 57.14%).



Sample Input and Output

Input	Output	Input	Output
3	66.67%	**4**	75.00%
1	0.00%	53	0.00%
2	0.00%	7	0.00%
999	0.00%	56	0.00%
	33.33%	999	25.00%

Input	Output	Input	Output
7	14.29%	**9**	33.33%
800	28.57%	367	33.33%
801	14.29%	99	11.11%
250	14.29%	200	11.11%
199	28.57%	799	11.11%
399		999	
599		333	
799		555	
		111	
		9	



Input	Output
14	57.14%
53	14.29%
7	7.14%
56	14.29%
180	7.14%
450	
920	
12	
7	
150	
250	
680	
2	
600	
200	

15. Smart Lilly

Lilly is **N years** old. For each **birthday** she receives a present. For each **odd** birthday (1, 3, 5, …, n) she receives **toys**, and for each **even** birthday (2, 4, 6, …, n) she receives **money**. For her **second birthday** she received **10.00 USD**, and **the amount is increased by 10.00 USD for each following even birthday** (2 -> 10, 4 -> 20, 6 -> 30 etc.). Over the years Lilly has secretly saved her money. Lilly's **brother**, in the years when she **received money**, **took 1.00 USD** from each of the amounts. Lilly **has sold the toys**, received over the years, **each one for P USD** and added the sum to the amount of saved money. With the money she wanted to **buy a washing machine for X USD**. Write a program that calculates **how much money she has saved** and if it is enough **to buy a washing machine**.

Input Data

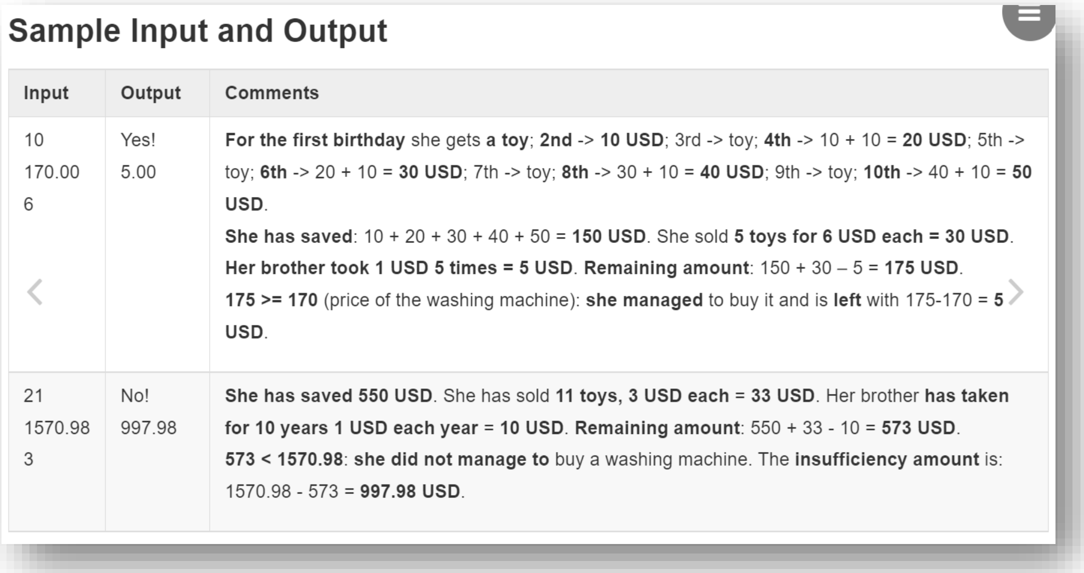
We read from the console **3 numbers**, each on a separate line:

- Lilly's **age** – **integer** in the range of [**1 … 77**].
- **Price of the washing machine** – number in the range of [**1.00 … 10 000.00**].
- **Unit price of each toy** – **integer** in the range of [**0 … 40**].

Output Data

Print on the console one single line:

- If Lilly's money is enough:
- "**Yes! {N}**" – where **N** is the remaining money after the purchase
- If the money is not enough:
- "**No! {M}**" – where **M** is the insufficiency amount
- Numbers **N** and **M** must be **formatted up to the second digit after the decimal point**.



Sample Input and Output

Input	Output	Comments
10 170.00 6	Yes! 5.00	**For the first birthday** she gets **a toy**; **2nd** -> **10 USD**; 3rd -> toy; **4th** -> 10 + 10 = **20 USD**; 5th -> toy; **6th** -> 20 + 10 = **30 USD**; 7th -> toy; **8th** -> 30 + 10 = **40 USD**; 9th -> toy; **10th** -> 40 + 10 = **50 USD**. **She has saved**: 10 + 20 + 30 + 40 + 50 = **150 USD**. She sold **5 toys for 6 USD each = 30 USD**. **Her brother took 1 USD 5 times = 5 USD**. **Remaining amount**: 150 + 30 – 5 = **175 USD**. **175 >= 170** (price of the washing machine): **she managed** to buy it and is **left** with 175-170 = **5 USD**.
21 1570.98 3	No! 997.98	**She has saved 550 USD**. She has sold **11 toys, 3 USD each = 33 USD**. Her brother **has taken for 10 years 1 USD each year = 10 USD**. **Remaining amount**: 550 + 33 - 10 = **573 USD**. **573 < 1570.98**: **she did not manage to** buy a washing machine. The **insufficiency amount** is: 1570.98 - 573 = **997.98 USD**.

16.Back to the Past

Ivan is **18 years old** and receives an inheritance that consists of **X** money and a **time machine**. He decides **to return to 1800**, but does not know **if the money** will **be enough** to live without working. Write **a program that calculates** if Ivan **will have enough money** to not have to work until a particular year (inclusive). Assuming that **for every even** (1800, 1802, etc.) year he **will spend 12 000 dollars**. For **every odd one** (1801, 1803, etc.) he will spend **12 000 + 50 * [the age he will have reached in the given year]**.

Input Data

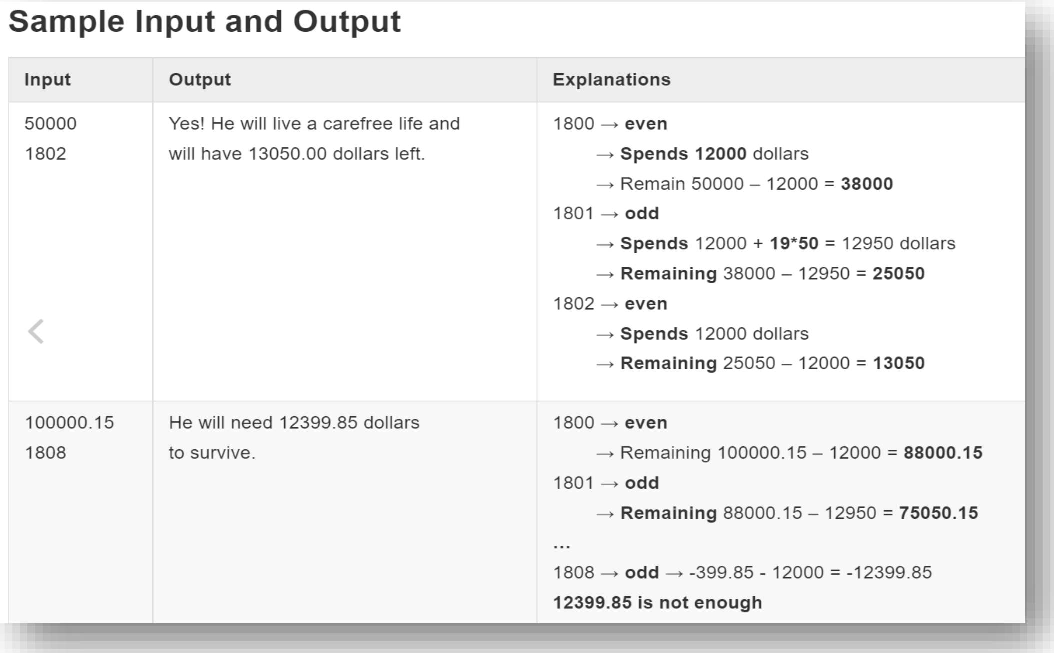
The input is read from the console and **contains exactly 2 lines**:

- **Inherited money** – a real number in the range [**1.00 … 1 000 000.00**].
- **Year, until which he has to live in the past (inclusive)** – integer number in the range [**1801 … 1900**].

Output Data

Print on the console **1 line**. The **sum** must be **formatted** up to the **two symbols after the decimal point**:

- If **money is enough**:
  - **"Yes! He will live a carefree life and will have {N} dollars left."** – where **N** is the money that will remain.
- If **money is NOT enough**:
  - **"He will need {M} dollars to survive."** – where **M** is the sum that **is NOT enough**.



Sample Input and Output

Input	Output	Explanations
50000 1802	Yes! He will live a carefree life and will have 13050.00 dollars left.	1800 → **even** → **Spends 12000** dollars → Remain 50000 – 12000 = **38000** 1801 → **odd** → **Spends** 12000 + **19*50** = 12950 dollars → **Remaining** 38000 – 12950 = **25050** 1802 → **even** → **Spends** 12000 dollars → **Remaining** 25050 – 12000 = **13050**
100000.15 1808	He will need 12399.85 dollars to survive.	1800 → **even** → Remaining 100000.15 – 12000 = **88000.15** 1801 → **odd** → **Remaining** 88000.15 – 12950 = **75050.15** ... 1808 → **odd** → -399.85 - 12000 = -12399.85 **12399.85 is not enough**

17.Hospital

For a certain period of time, patients arrive at the hospital every day for an examination. It has **initially 7 doctors**. Each doctor can treat only **one patient per day**, but sometimes there is a shortage of doctors, so the **remaining patients are sent to other hospitals**. **Every third day** the hospital makes calculations and **if the count of untreated patients is greater than the count of treated ones, another doctor is appointed**. Appointment takes place before the daily patient acceptance begins.

Write a program, that calculates **for a given period of time, the count of treated and untreated patients**.

Input Data

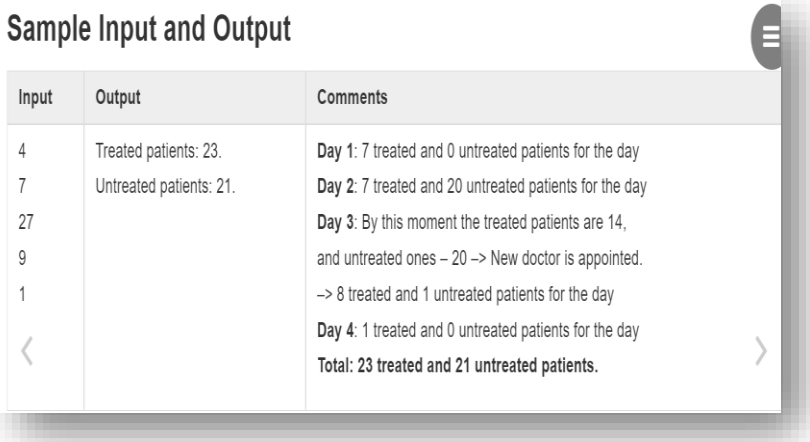
Input is read from the **console** and contains:

- On the first line – **the period**, for which you need to make calculations. **Integer** in the range of [**1 … 1000**].
- On the next lines (equal to the count of days) – **count of the patients**, who arrive for treatment for the **current day**. Integer in the range of [**0 … 10 000**].

Output Data

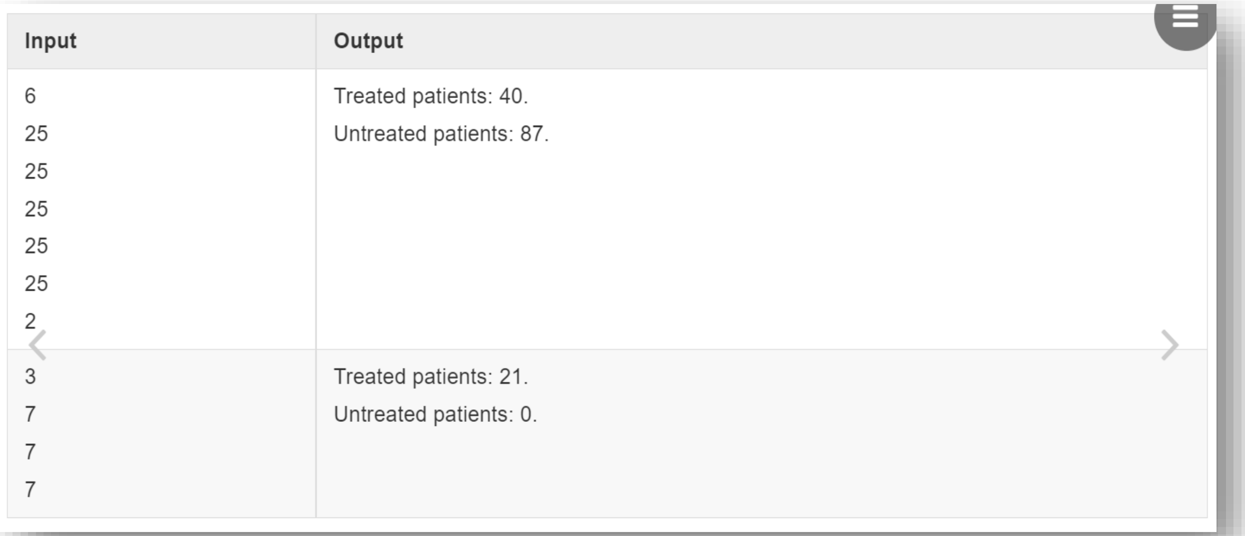
Print **on the console 2 lines**:

- On the **first line**: “**Treated patients: {count of treated patients}.**”
- On the **second line**: “**Untreated patients: {count of untreated patients}.**”



Sample Input and Output

Input	Output	Comments
4 7 27 9 1	Treated patients: 23. Untreated patients: 21.	**Day 1**: 7 treated and 0 untreated patients for the day **Day 2**: 7 treated and 20 untreated patients for the day **Day 3**: By this moment the treated patients are 14, and untreated ones – 20 –> New doctor is appointed. –> 8 treated and 1 untreated patients for the day **Day 4**: 1 treated and 0 untreated patients for the day **Total: 23 treated and 21 untreated patients.**



Input	Output
6 25 25 25 25 25 2	Treated patients: 40. Untreated patients: 87.
3 7 7 7	Treated patients: 21. Untreated patients: 0.

18.Division without Remainder

We have **n integers** in the range of [**1 ... 1000**]. Among them, **some percentage p1 are divisible without remainder by 2**, **percentage p2** are **divisible without remainder by 3**, **percentage p3** are **divisible without remainder by 4**. Write a program that calculates and prints the p1, p2 and p3 percentages. **Example:** We have **n = 10** numbers: 680, 2, 600, 200, 800, 799, 199, 46, 128, 65. We get the following distribution and visualization:

Division without remainder by:	**Numbers**	**Count**	**Percent**
2	680, 2, 600, 200, 800, 46, 128	7	p1 = (7 / 10) * 100 = **70.00%**
3	600	1	p2 = (1 / 10) * 100 = **10.00%**
4	680, 600, 200, 800, 128	5	p3 = (5 / 10) * 100 = **50.00%**

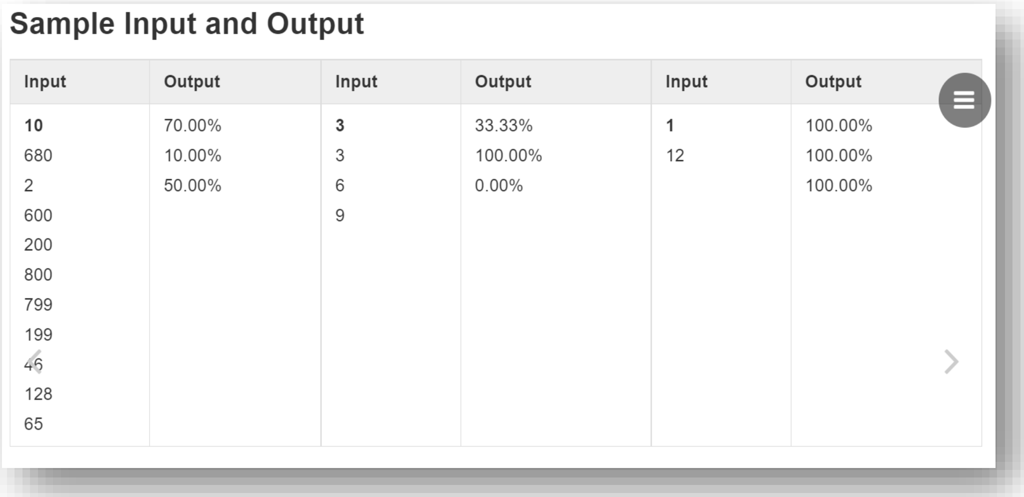
Input Data

On the first line of the input is the integer **n** (1 ≤ **n** ≤ 1000) – count of numbers. On each of the next **n lines** we have **one integer** in the range of [**1 … 1000**] – numbers that needs to be checked for division.

Output Data

Print on the console **3 lines**, each of them containing a percentage between 0% and 100%, two digits after the decimal point, for example 25.00%, 66.67%, 57.14%.

- On the **first line** – percentage of the numbers that are **divisible by 2**.
- On the **second line** – percentage of the numbers that are **divisible by 3**.
- On the **third line** – percentage of the numbers that are **divisible by 4**.



Sample Input and Output

Input	Output	Input	Output	Input	Output
10 680 2 600 200 800 799 199 46 128 65	70.00% 10.00% 50.00%	**3** 3 6 9	33.33% 100.00% 0.00%	**1** 12	100.00% 100.00% 100.00%

19.Logistics

You are responsible for the logistics various types of cargo. **Depending on the weight** of each cargo, you need a **different vehicle**, and this will cost a **different price per ton**:

- Up to **3 tons** – a **minibus** (200 USD per ton).
- From **over 3 and up to 11 tons** – **truck** (175 USD per ton).
- **Over 11 tons – train** (120 USD per ton).

Your task is to calculate **the average price per ton of the cargo**, and also **what percentage of the cargo is transported in each vehicle**.

Input Data

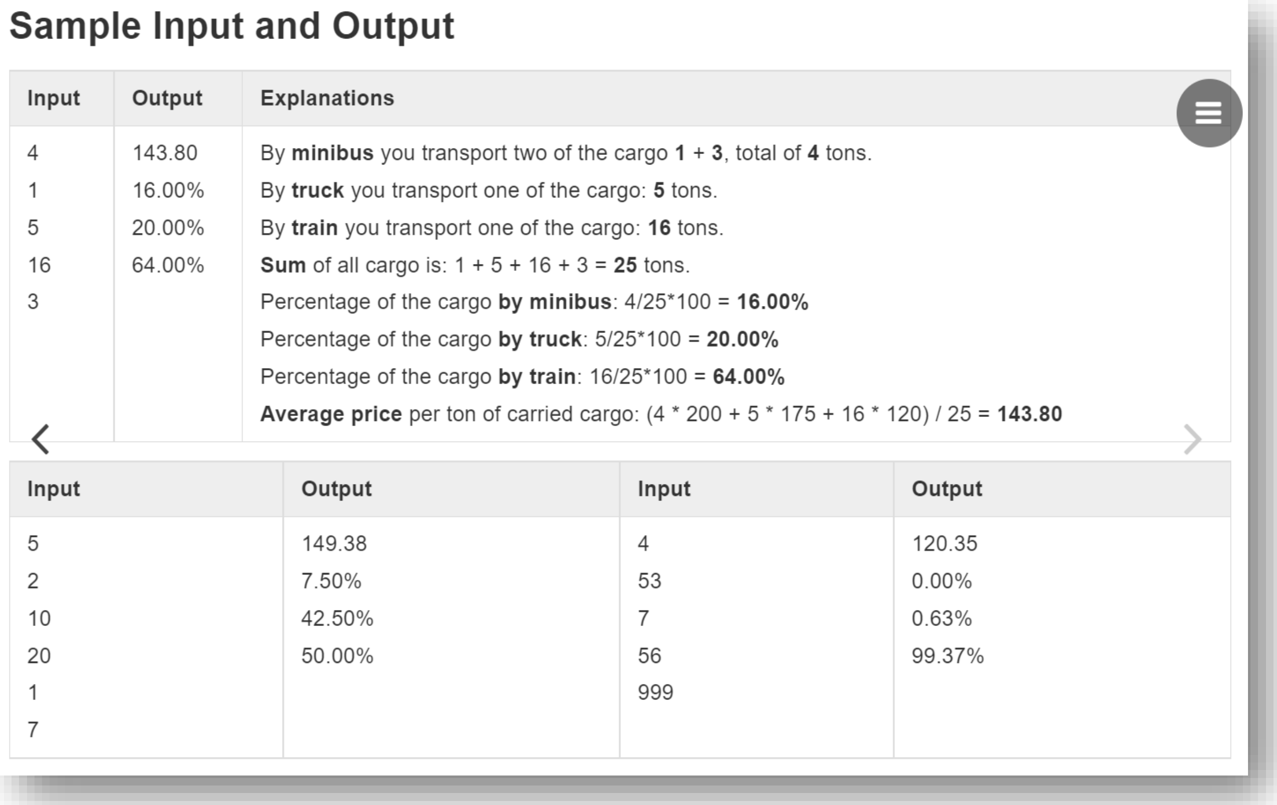
From the console we must read a **sequence of numbers**, each on a separate line:

- **First line**: **count of cargo** for transportation – **integer** in the range of [**1 … 1000**].
- On the next lines we pass **the tonnage of the current cargo** – **integer** in the range of [**1 … 1000**].

Output Data

Print on the console **4 lines**, as follows:

- **Line #1** – **the average price per ton of the cargo** (rounded up to the second digit after the decimal point).
- **Line #2** – **percentage** of the cargo, carried by **minibus** (between 0.00% and 100.00%, rounded up to the second digit after the decimal point).
- **Line #3** – **percentage** of the cargo, carried by **truck** (between 0.00% and 100.00%).
- **Line #4** – **percentage** of the cargo, carried by **train** (between 0.00% and 100.00%).

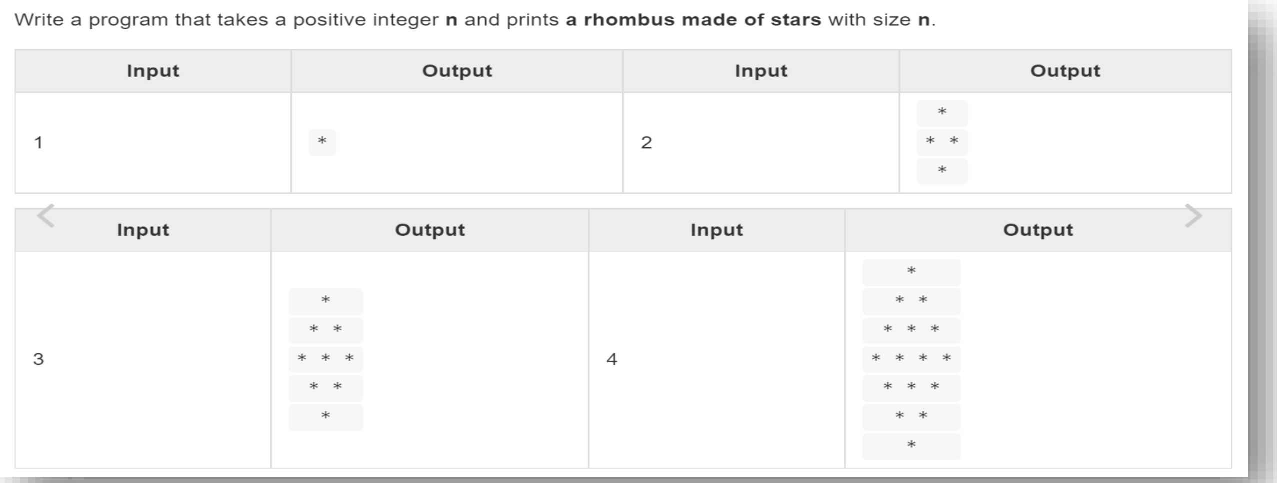


Sample Input and Output

Input	Output	Explanations
4 1 5 16 3	143.80 16.00% 20.00% 64.00%	By **minibus** you transport two of the cargo **1** + **3**, total of **4** tons. By **truck** you transport one of the cargo: **5** tons. By **train** you transport one of the cargo: **16** tons. **Sum** of all cargo is: 1 + 5 + 16 + 3 = **25** tons. Percentage of the cargo **by minibus**: 4/25*100 = **16.00%** Percentage of the cargo **by truck**: 5/25*100 = **20.00%** Percentage of the cargo **by train**: 16/25*100 = **64.00%** **Average price** per ton of carried cargo: (4 * 200 + 5 * 175 + 16 * 120) / 25 = **143.80**

Input	Output	Input	Output
5 2 10 20 1 7	149.38 7.50% 42.50% 50.00%	4 53 7 56 999	120.35 0.00% 0.63% 99.37%

20.Rhombus Made of Stars



Write a program that takes a positive integer **n** and prints **a rhombus made of stars** with size **n**.

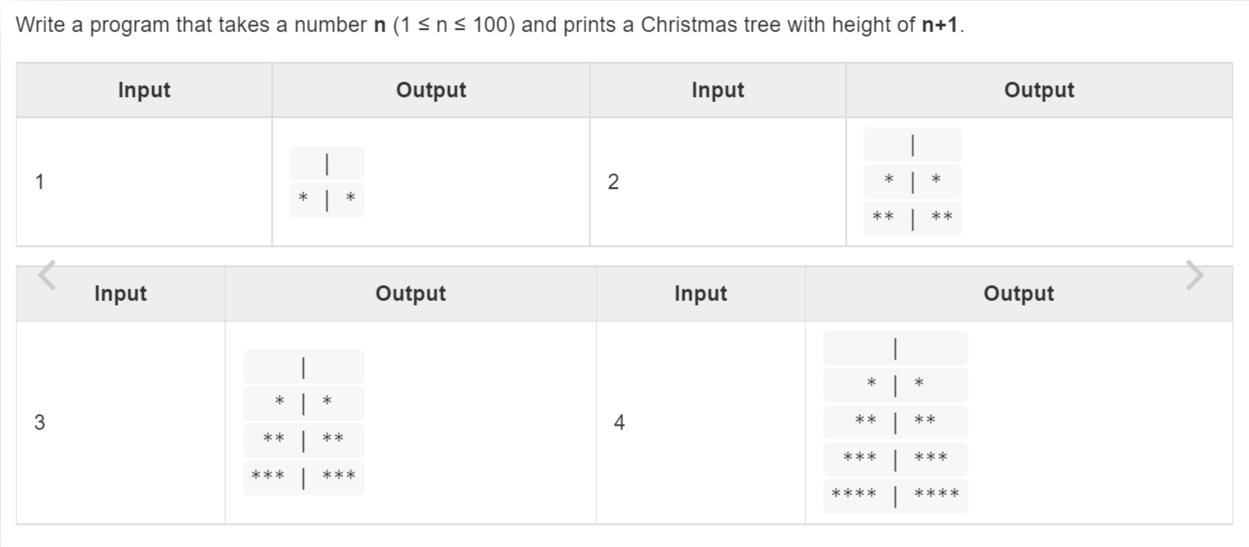
Input	Output	Input	Output

1	`*`	2	` *`
			`* *`
			` *`

Input	Output	Input	Output

3	`  *`
	` * *`
	`* * *`
	` * *`
	`  *`
		4	`   *`
			`  * *`
			` * * *`
			`* * * *`
			` * * *`
			`  * *`
			`   *`

21.Christmas Tree

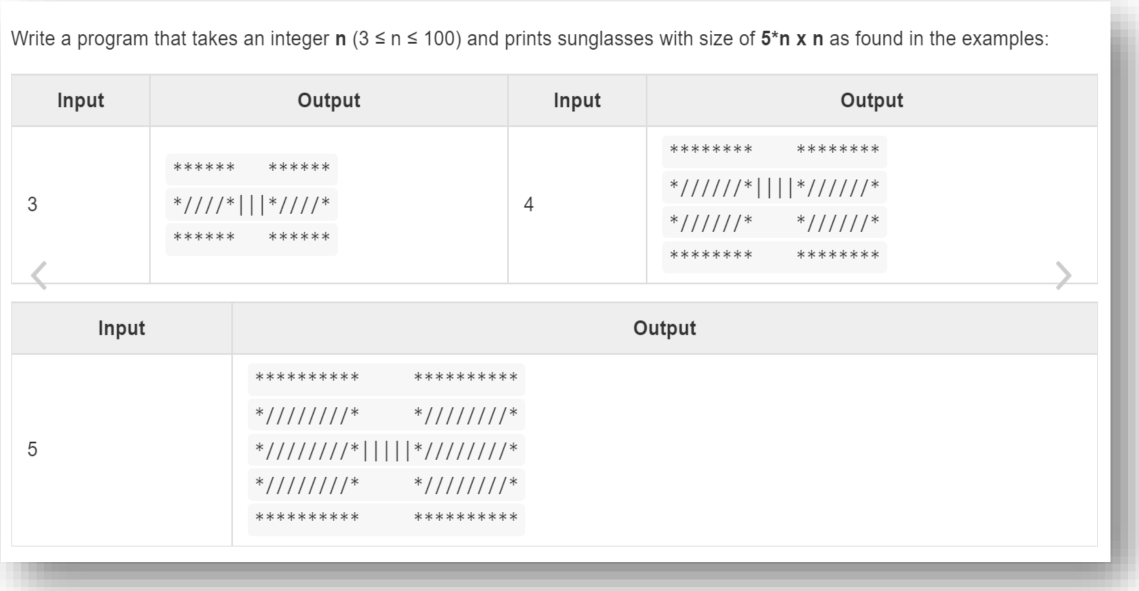


Write a program that takes a number **n** (1 ≤ n ≤ 100) and prints a Christmas tree with height of **n+1**.

<table>
<tr><th>Input</th><th>Output</th><th>Input</th><th>Output</th></tr>
<tr><td>1</td><td> |
* | *</td><td>2</td><td> |
 * | *
** | **</td></tr>
</table>

<table>
<tr><th>Input</th><th>Output</th><th>Input</th><th>Output</th></tr>
<tr><td>3</td><td> |
  * | *
 ** | **
*** | ***</td><td>4</td><td> |
   * | *
  ** | **
 *** | ***
**** | ****</td></tr>
</table>

22.Sunglasses

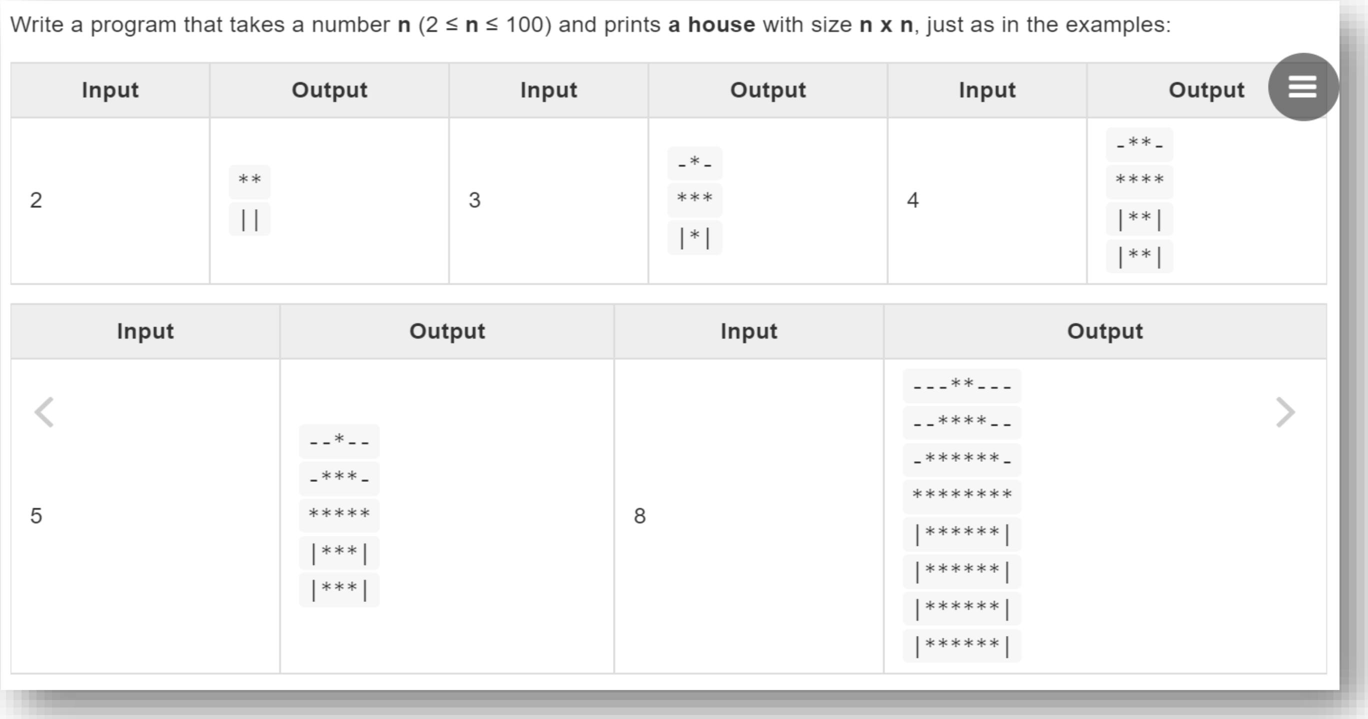


Write a program that takes an integer **n** (3 ≤ n ≤ 100) and prints sunglasses with size of **5*n x n** as found in the examples:

<table>
<tr><th>Input</th><th>Output</th><th>Input</th><th>Output</th></tr>
<tr><td>3</td><td>******   ******
////|||*////*
******   ******</td><td>4</td><td>********   ********
//////||||*//////*
//////   *//////*
********   ********</td></tr>
</table>

<table>
<tr><th>Input</th><th>Output</th></tr>
<tr><td>5</td><td>**********   **********
////////   *////////*
////////|||||*////////*
////////   *////////*
**********   **********</td></tr>
</table>

23.House

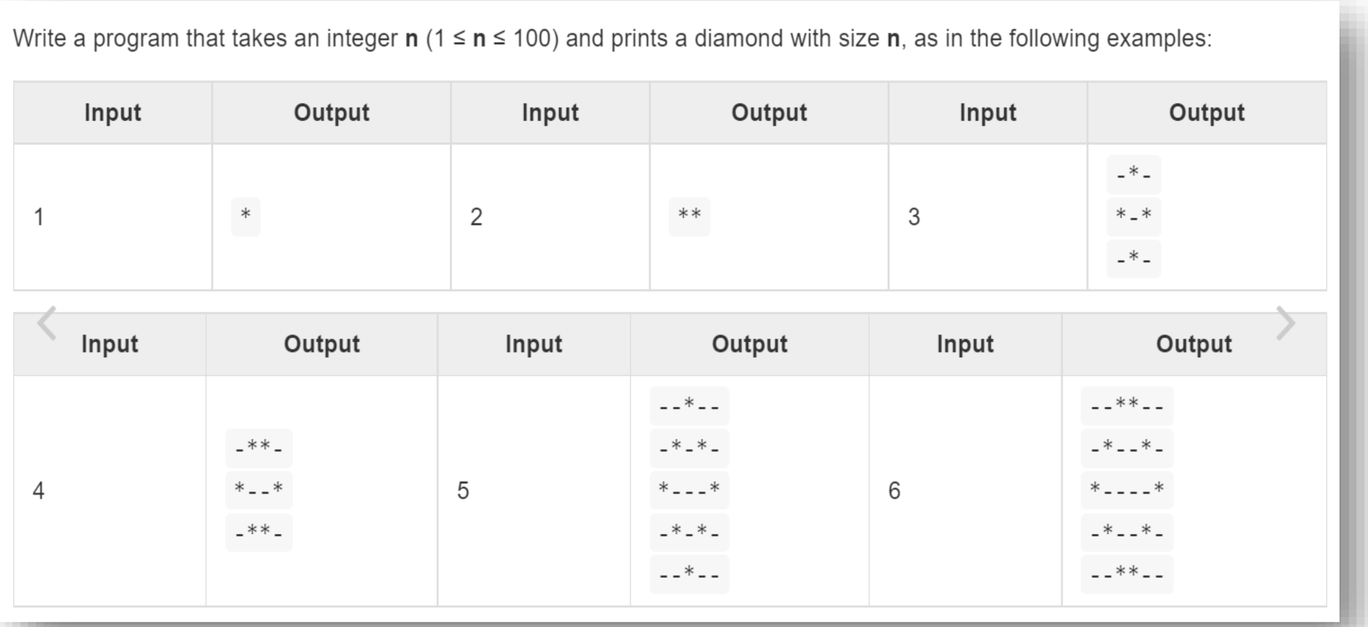


Write a program that takes a number **n** (2 ≤ **n** ≤ 100) and prints **a house** with size **n x n**, just as in the examples:

<table>
<tr><th>Input</th><th>Output</th><th>Input</th><th>Output</th><th>Input</th><th>Output</th></tr>
<tr><td>2</td><td>**
||</td><td>3</td><td>-*-
***
|*|</td><td>4</td><td>-**-
****
|**|
|**|</td></tr>
</table>

<table>
<tr><th>Input</th><th>Output</th><th>Input</th><th>Output</th></tr>
<tr><td>5</td><td>--*--
-***-
*****
|***|
|***|</td><td>8</td><td>---**---
--****--
-******-
****
|******|
|******|
|******|
|******|</td></tr>
</table>

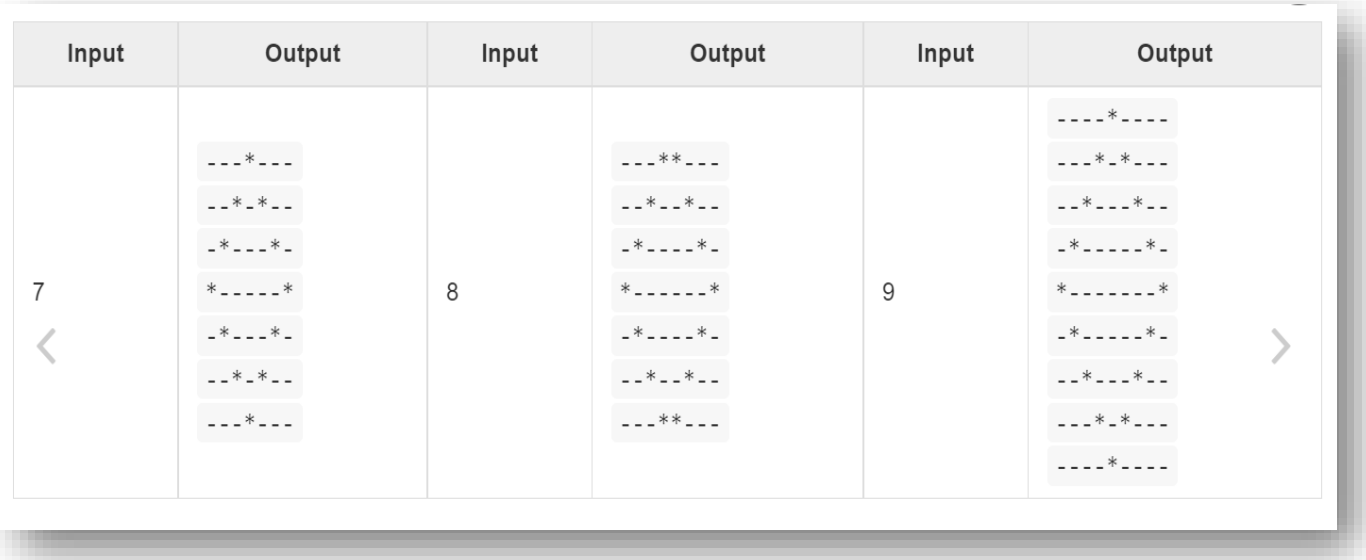
24.Diamond



Write a program that takes an integer **n** (1 ≤ **n** ≤ 100) and prints a diamond with size **n**, as in the following examples:

Input	Output	Input	Output	Input	Output
1	*	2	**	3	-*- *-* -*-

Input	Output	Input	Output	Input	Output
4	-**- *--* -**-	5	--*-- -*-*- *---* -*-*- --*--	6	--**-- -*--*- *----* -*--*- --**--



Input	Output	Input	Output	Input	Output
7	---*--- --*-*-- -*---*- *-----* -*---*- --*-*-- ---*---	8	---**--- --*--*-- -*----*- *------* -*----*- --*--*-- ---**---	9	----*---- ---*-*--- --*---*-- -*-----*- *-------* -*-----*- --*---*-- ---*-*--- ----*----