Exercise: Basic Syntax, Conditional Statements and Loops

Problems for exercises and homework for the "Programming Fundamentals" course @ SoftUni.

You can check your solutions in Judge.

1. Ages

Write a program that determines whether a person is based on the given age: baby, child, teenager, adult, or elder. The bounders are:

- **0-2 – baby;**
- **3-13 – child;**
- **14-19 – teenager;**
- **20-65 – adult;**
- **>=66 – elder;**
- All the values are **inclusive**.

Examples

Input	Output
20	adult
1	baby
100	elder

2. Division

You will be given an integer, and you have to print on the console whether that number is divisible by the following numbers: 2, 3, 6, 7, 10. You should **always take the bigger division**. If the number is divisible by both **2** and **3** it is also divisible by **6**, and you should print only the division by 6. If a number is divisible by **2** it is sometimes also divisible by **10**, and you should print the division by 10. If the number is not divisible by any given number, print "**`Not divisible`**". Otherwise, print "**`The number is divisible by {number}`**".

Examples

Input	Output
30	The number is divisible by 10
15	The number is divisible by 3
12	The number is divisible by 6
1643	Not divisible

3. Vacation

You are given a group of people, type of the group, on which day of the week they will stay. Based on that information, calculate how much they must pay and print that price on the console. Use the table below. In each cell is the price for a **single person**. The output should look like that: "**`Total price: {price}`**". The price should be formatted to the second decimal point.

	Friday	Saturday	Sunday
Students	8.45	9.80	10.46
Business	10.90	15.60	16
Regular	15	20	22.50

There are also discounts based on some conditions:

- **Students –** if the group is bigger than or equal to 30 people, you should reduce the **total** price by 15%
- **Business –** if the group is bigger than or equal to 100 people **10** of them can stay **for free.**
- **Regular –** if the group is bigger than or equal to 10 and less than or equal to 20 reduce the **total** price by 5%

You should reduce the prices in that EXACT order.

Examples

Input	Output
30 Students Sunday	Total price: 266.73
40 Regular Saturday	Total price: 800.00
50 Business Friday	Total price: 545.00

4. Print and Sum

Write a program to display numbers from given start to given end and their sum. All the numbers will be integers. On the first line, you will receive the start number, on the second the end number.

Examples

Input	Output
5 10	5 6 7 8 9 10 Sum: 45
0 26	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 Sum: 351
50 60	50 51 52 53 54 55 56 57 58 59 60 Sum: 605

5. Login

You will be given a string representing a username. The password will be that username reversed. Until you receive the correct password, print on the console "**`Incorrect password. Try again.`**". When you receive the correct password, print "**`User {username} logged in.`**" However, on the fourth try, if the password is still not correct, print "**`User {username} blocked!`**" and end the program.

Examples

Input	Output
Acer login go let me in recA	Incorrect password. Try again. Incorrect password. Try again. Incorrect password. Try again. User Acer logged in.
momo omom	User momo logged in.
sunny rainy cloudy sunny not sunny	Incorrect password. Try again. Incorrect password. Try again. Incorrect password. Try again. User sunny blocked!

6. Strong Number

Write a program to check whether or not a given number is strong. A number is strong if the sum of the Factorial of each digit is equal to the number. For example 145 is a strong number, because **1! + 4! + 5! = 145.** Print "**yes**" if the number is strong and "**no**" if the number is not strong.

Examples

Input	Output
2	yes
3451	no
40585	yes

7. Vending Machine

Your task is to calculate the total purchase price from a vending machine. Until you receive "**`Start`**" you will be given different coins that are being inserted into the machine. You have to sum them to have the total money inserted. There is a problem though. Your vending machine only works with **0.1**, **0.2**, **0.5, 1, and 2** coins. If someone tries to insert some other coins, you have to display "**`Cannot accept {money}`**", where the value is **formatted to the second digit after the decimal point** and **not** add it to the total money. On the next few lines until you receive "**End**" you will be given products to purchase. Your machine has, however, only "**`Nuts`**", "**`Water`**", "**`Crisps`**", "**Soda**", "**Coke**". The prices are: **`2.0`**, **`0.7`**, **`1.5`**, **`0.8`**, **`1.0`** respectively. If the person tries to purchase a not existing product, print "**`Invalid product`**". Be careful that the person may try to purchase a product for which he doesn't have money. In that case, print "**`Sorry, not enough money`**". If the person purchases a product successfully

print "**Purchased {product name}**". After the "**End**" command, print the money that is left formatted to the second decimal point in the format "**Change: {money left}**".

Examples

Input	Output
1 1 0.5 0.6 Start Coke Soda Crisps End	Cannot accept 0.60 Purchased Coke Purchased Soda Sorry, not enough money Change: 0.70
1 Start Nuts Coke End	Sorry, not enough money Purchased Coke Change: 0.00

8. Triangle of Numbers

Write a program that receives a number – **n** and prints a triangle from **1 to n** as in the examples.

Constraints

- **n** will be in the interval **[1...20]**.

Examples

Input	Output
3	1 2 2 3 3 3

Input	Output
5	1 2 2 3 3 3 4 4 4 4 5 5 5 5 5

Input	Output
6	1 2 2 3 3 3 4 4 4 4 5 5 5 5 5 6 6 6 6 6 6

9. Orders

We are placing **N** orders at a time. You need to calculate the price on the following formula:

((daysInMonth * capsulesCount) * pricePerCapsule)

Input / Constraints

- On the first line, you will receive integer **N** – the count of orders the shop will receive.
- For each order, you will receive the following information:
  - Price per capsule - **floating-point number** in the range **[0.00...5000.00].**
  - Days – **integer** in the range **[1...31].**
  - Capsules count - **integer** in the range **[0...7000000].**

The input will be in the described format, there is no need to check it explicitly.

Output

The output should consist of **N + 1** line. For each order, you must print a single line in the following format:

- **`"The price for the coffee is: ${price}"`**

On the last line, you need to print the total price in the following format:

- **`"Total: ${totalPrice}"`**

The **price must be formatted** to 2 decimal places.

Examples

Input	Output	Comments
1 1.53 30 8	The price for the coffee is: $367.20 Total: $367.20	We are given only 1 order. Then we use the formulas: **orderPrice** = 30 * 8 * 1.53 = 367.20
2 4.99 31 3 0.35 31 5	The price for the coffee is: $464.07 The price for the coffee is: $54.25 Total: $518.32	

10. *Padawan Equipment

Yoda is starting his newly created Jedi academy. So, he asked Master George Lucas to **buy** the **needed equipment**. The number of **items** depends on **how many students will sign up**. The equipment for the Padawan contains **lightsabers, belts, and robes**.

You will be given **the amount of money George Lucas has**, the **number of students,** and the **prices of each item**. You have to help George Lucas **calculate** if the **money** he has is **enough to buy all of the equipment** or how much more money he needs.
Because the lightsabers sometimes break, George Lucas should **buy 10% more**, **rounded up** to the next integer. Also, every **sixth belt is free**.

Input / Constraints

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

- The **amount of money** George Lucas has – the **floating-point number** in the **range [0.00…1,000.00].**
- The **count of students – integer in the range [0…100].**
- The **price of lightsabers** for a **single saber – the floating-point number** in the **range [0.00…100.00].**
- The **price of robes** for a **single robe – the floating-point number** in the **range [0.00…100.00].**
- The **price of belts** for a **single belt – the floating-point number** in the **range [0.00…100.00].**

The **input data will always be valid**. **There is no need to check it explicitly**.

Output

The output should be printed on the console.

- **If the calculated price of the equipment is less or equal to the money George Lucas has:**
  "The money is enough - it would cost {the cost of the equipment}lv."
- **If the calculated price of the equipment is more than the money George Lucas has:**
  "George Lucas will need {neededMoney}lv more."
- **All prices** must be **rounded to two digits after the decimal point.**

Examples

Input	Output	Comments
100 2 1.0 2.0 3.0	The money is enough - it would cost 13.00lv.	Needed equipment for **2** padawans: **sabresPrice** * (**studentsCount** + 10%) + **robesPrice** * (**studentsCount**) + **beltsPrice** * (**studentsCount - freeBelts**) 1*(3) + 2*(2) + 3*(2) = 13.00 13.00 <= 100 – the money will be enough.
100 42 12.0 4.0 3.0	George Lucas will need 737.00lv more.	Needed equipment for **42** padawans: 12*47 + 4*42 + 3*35 = 837.00 837 > 100 – need 737.00 lv. more.

11. *Rage Expenses

As a MOBA challenger player, Peter has the bad habit of trashing his PC when he loses a game and rage quits. His gaming setup consists of a **headset, mouse, keyboard, and display**. You will receive Peter's **lost games count**.

Every **second** lost game, Peter trashes his **headset.**

Every **third** lost game, Peter trashes his **mouse**.

When Peter trashes **both his mouse and headset** in the **same** lost game, he also trashes his **keyboard**.

Every second time when he trashes his keyboard, he also trashes his **display**.

You will receive the price of each item in his gaming setup. Calculate his rage expenses for renewing his gaming equipment.

Input / Constraints

- On the first input line - **lost games count** – integer in the range **[0, 1000]**.
- On the second line – **headset price** - the floating-point number in the range **[0, 1000]**.
- On the third line – **mouse price** - the floating-point number in the range **[0, 1000]**.
- On the fourth line – **keyboard price** - the floating-point number in the range **[0, 1000]**.
- On the fifth line – **display price** - the floating-point number in the range **[0, 1000]**.

Output

- As output you must print Peter's total expenses: **"Rage expenses: {expenses} lv."**
- Allowed working **time** / **memory**: **100ms** / **16MB**.

Examples

Input	Output	Comment

7 2 3 4 5	Rage expenses: 16.00 lv.	Trashed headset -> **3** times Trashed mouse -> **2** times Trashed keyboard -> **1** time Total: 6 + 6 + 4 = 16.00 lv;
23 12.50 21.50 40 200	Rage expenses: 608.00 lv.	