Exercises: Delegates and Events

You can check your solutions in **Judge system**: https://judge.softuni.bg/Contests/3168/Delegates-and-Events

1. Sort Even Numbers

Write a program that reads one line of **integers** separated by **", "**. Then prints the **even numbers** of that sequence **sorted** in **increasing** order.

Examples

Input	Output
4, 2, 1, 3, 5, 7, 1, 4, 2, 12	2, 2, 4, 4, 12

Input	Output
1, 3, 5	

Input	Output
2, 4, 6	2, 4, 6

Hint

It is up to you what type of data structures you will use to solve this problem. **Use functional programming filter** and sort the collection of numbers.

2. Sum Numbers

Write a program that reads a line of **integers** separated by **", "**. Print on two lines the **count** of numbers and their **sum**. Use the built-in **agregate function** for summing.

Examples

Input	Output
4, 2, 1, 3, 5, 7, 1, 4, 2, 12	10 41
2, 4, 6	3 12

3. Count Uppercase Words

Write a program that reads a line of **text** from the console. Print **all** the words that start with an **uppercase letter** in the **same order** you've received them in the text.

Examples

Input	Output
The following example shows how to use Function	The Function
Write a program that reads one line of text from console. Print count of words that start with	Write Print

`Uppercase, after that print all those words in the same order like you find them in text.`	`Uppercase,`

Hint

Use **`Func<string, bool>`** for filtering the input words and use **" "** for splitting words.

4. Add VAT

Write a program that reads one line of **double** prices separated by **", "**. Print the **prices** with **added VAT** for all of them. **Format** them to **2 signs** after the decimal point. The **order** of the prices must be the **same**.
VAT is equal to 20% of the price. Use a **function double → double** to calculate the VAT.

Examples

Input	Output

`1.38, 2.56, 4.4`	`1.66`
	`3.07`
	`5.28`

Input	Output

`1, 3, 5, 7`	`1.20`
	`3.60`
	`6.00`
	`8.40`

5. Filter by Age

Write a program that receives an integer **N** on first line. On the next **N** lines, read pairs of **"`[name], [age]`"**. Then read three lines with:

- **Condition** – "**`younger`**" or "**`older`**"
- **Age** – Integer
- **Format** – "**name**", "**age**" or "**name age**"

Depending on the **condition**, print the correct **pairs** in the correct **format**. **Don't use the built-in functionality from .NET. Create your own methods**.

Examples

Input	Output

`5`
`Lucas, 20`
`Tomas, 18`
`Mia, 29`
`Noah, 31`
`Simo, 16`
`older`
`20`
`name age`
	`Lucas - 20`
	`Mia - 29`
	`Noah - 31`

Input	Output

`5`
`Lucas, 20`
`Tomas, 18`
`Mia, 29`
`Noah, 31`
`Simo, 16`
`younger`
`20`
`name`
	`Tomas`
	`Simo`

Input	Output

`5`
`Lucas, 20`
`Tomas, 18`
`Mia, 29`
`Noah, 31`
`Simo, 16`
`younger`
`50`
`age`
	`20`
	`18`
	`29`
	`31`
	`16`

6. Action Print

Write a program that reads a collection of **strings** from the console and then **prints** them onto the **console**. Each name should be printed on a **new line**. Use **`Action<T>`**.

Examples

Input	Output
Lucas Noah Tea	Lucas Noah Tea

7. Knights of Honor

Write a program that reads a collection of **names** as **strings** from the **console**, appends "**Sir**" in front of every name and **prints** it back on the **console**. Use **Action<T>**.

Examples

Input	Output
Eathan Lucas Noah StanleyRoyce	Sir Eathan Sir Lucas Sir Noah Sir StanleyRoyce

8. Custom Min Function

Write a simple program that reads from the **console** a set of **integers** and **prints** back on the **console** the **smallest number** from the collection. Use **Func<T, T>**.

Examples

Input	Output
1 4 3 2 1 7 13	1

9. Find Evens or Odds

You are given a lower and an upper bound for a range of integer numbers. Then a command specifies if you need to list all even or odd numbers in the given range. Use **Predicate<T>**.

Examples

Input	Output
1 10 odd	1 3 5 7 9
20 30 even	20 22 24 26 28 30

10. Reverse and Exclude

Write a program that reverses a collection and removes elements that are divisible by a given integer **n**. Use **predicates/functions**.

Examples

Input	Output

`1 2 3 4 5 6`
`2`
	`5 3 1`

`20 10 40 30 60 50`
`3`
	`50 40 10 20`

11. Predicate for Names

Write a program that filters a list of names according to their length. On the first line, you will be given an integer **n,** representing a name's length. On the second line, you will be given some names as strings separated by space. Write a function that prints only the names whose length is **less than or equal** to **n**.

Examples

Input	Output

`4`
`Nathaniel Molly Dan Joe Glen`
	`Dan`
	`Joe`
	`Glen`

`4`
`Alexia Cara Robin Lynda`
	`Cara`

12. Custom Comparator

Write a custom comparator that sorts all even numbers before all the odd ones in ascending order. Pass it to **`Array.Sort()`** function and print the result. Use functions.

Examples

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

13. List of Predicates

Find all numbers in the range 1...N that are divisible by the numbers of a given sequence. On the first line, you will be given an integer **N** – which is the end of the range. On the second line, you will be given a sequence of integers which are the dividers. Use predicates/functions.

Examples

Input	Output

`10`
`1 1 1 2`
	`2 4 6 8 10`

`100`
`2 5 10 20`
	`20 40 60 80 100`

14. Predicate Party!

Carlos's parents are on a vacation for the holidays and he is planning an epic party at home. Unfortunately, his organizational skills are next to non-existent, so you are given the task to help him with the reservations.

On the **first line,** you receive a **list with all the people** that are coming. On the **next lines**, until you get the **"Party!" command**, you may be asked to **double** or **remove all the people** that apply to a given **criteria**. There are **three different criteria**:

- Everyone that has his **name starting** with a **given string**
- Everyone that has a **name ending** with a **given string**
- Everyone that has a **name** with a **given length**.

Finally, **print all the guests** who are going to the party **separated by** ", " and then **add the ending** "are going to the party!". If there are **no guests** going to the party print "Nobody is going to the party!". See the examples below:

Examples

Input	Output

`Paul Alice Hector`
`Remove StartsWith P`
`Double Length 5`
`Party!`
	`Alice, Alice, Hector are going to the party!`

`Peter`
`Double StartsWith Pete`
`Double EndsWith eter`
`Party!`
	`Peter, Peter, Peter, Peter are going to the party!`

`Philip`
`Remove StartsWith P`
`Party!`
	`Nobody is going to the party!`

15. Party Reservation Filter Module

You need to implement a filtering module to a party reservation software. First, to the Party Reservation Filter Module (PRFM for short) is **passed a list** with invitations. Next the PRFM receives a **sequence of commands** that specify whether you need to add or remove a given filter.

Each PRFM command is in the given format:

`"{command;filter type;filter parameter}"`

You can receive the following PRFM commands:

- **`"Add filter"`**
- **`"Remove filter"`**
- **`"Print"`**

The possible PRFM filter types are:

- **`"Starts with"`**
- **`"Ends with"`**
- **`"Length"`**
- **`"Contains"`**

All PRFM filter parameters will be a string (or an integer only for the "**`Length`**" filter). Each command will be valid e.g. you won't be asked to remove a non-existent filter. The input will **end** with a "**Print**" command, after which you should print all the party-goers that are left after the filtration. See the examples below:

Examples

Input	Output

`Paul Mark Sandy`
`Add filter;Starts with;P`
`Add filter;Starts with;M`
`Print`
	`Sandy`

`Peter Miles Jorge`
`Add filter;Starts with;P`
`Add filter;Starts with;M`
`Remove filter;Starts with;M`
`Print`
	`Miles Jorge`

16. Event Calculator

Write program that simply adds two numbers. You will receive **the numbers separated by a single space**, **print the sum** of the **numbers and** if is **odd number, fire** an **event** that prints the message **`"********Event Executed : This is Odd Number**********"`** using delegates.

Create **public** class **`AddTwoNumbers`**, declare in it delegate, event and create method **`Add()`**. The method must calculate the certain numbers and check whether the event is fired. In the **`StartUp`** class, create an object of the class **`AddTwoNumbers`**, subscribe it to the event and use the method **`Add()`**.

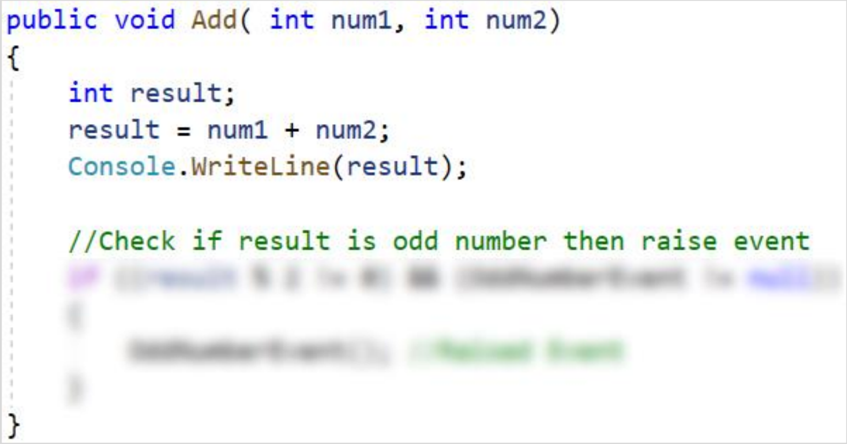
Examples

Input	Output

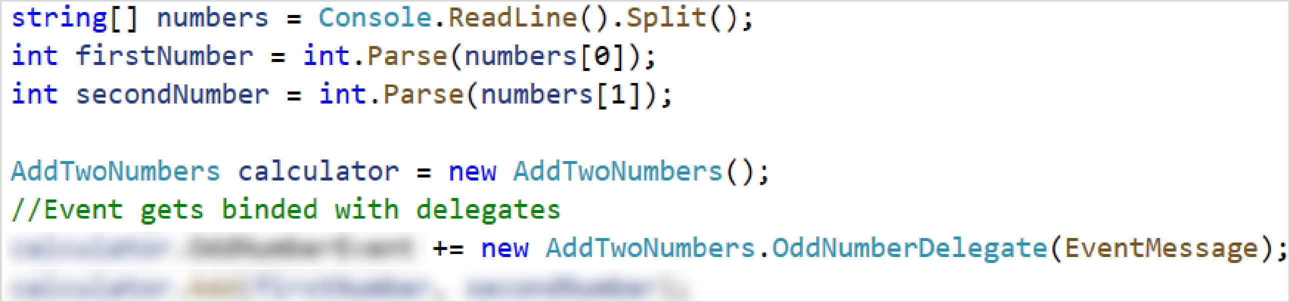
`3 4`	`7`
	`********Event Executed : This is Odd Number**********`

2 2	4

Hints:



```
public void Add( int num1, int num2)
{
    int result;
    result = num1 + num2;
    Console.WriteLine(result);

    //Check if result is odd number then raise event
    [redacted]
}
```



```
string[] numbers = Console.ReadLine().Split();
int firstNumber = int.Parse(numbers[0]);
int secondNumber = int.Parse(numbers[1]);

AddTwoNumbers calculator = new AddTwoNumbers();
//Event gets binded with delegates
[redacted] += new AddTwoNumbers.OddNumberDelegate(EventMessage);
```

17. Console Key Event

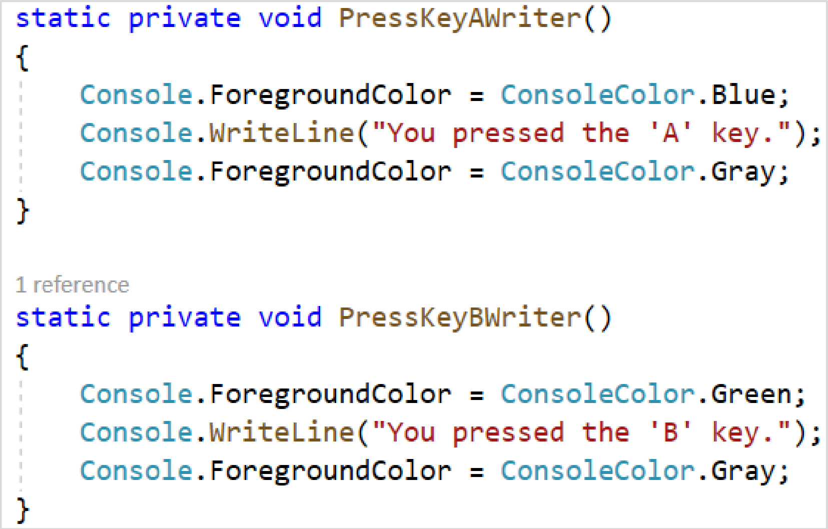
This problem cannot be tested in Judge!

Write a program that, when you press the `[a]` or `[b]` keyboard key, **fires** an **event** that uses a method to write in color on the console the following message: **`"You pressed the 'A' key."`**, **`"You pressed the 'B' key."`** or **`"No event handler for key {key}"`**.

To do this, create a **public** class **`Keyboard`** that contains the following members and methods:

- One **delegate `PressKeyEvent`**
- Two **events** equal to null for example **`PressKeyA`** and **`PressKeyB`**
- **`void PressKeyAEvent()`** – that if the event **`PressKeyA`** is not null, invokes it by the base system method **`.Invoke()`**
- **`void PressKeyBEvent()`** – that if the event **`PressKeyB`** is not null, invokes it by the base system method **`.Invoke()`**
- **`void Start()`** – **reads** from the console, **switches** if key `[a]` or `[b]` is pressed and calls the corresponding method. The **default** behavior (any other key is pressed) is to write on the console **`"No event handler for key {key}"`**

Create the following methods in the main method that you can give as arguments to the subscribed delegate:



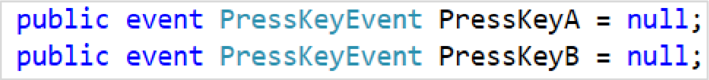
```
static private void PressKeyAWriter()
{
    Console.ForegroundColor = ConsoleColor.Blue;
    Console.WriteLine("You pressed the 'A' key.");
    Console.ForegroundColor = ConsoleColor.Gray;
}

1 reference
static private void PressKeyBWriter()
{
    Console.ForegroundColor = ConsoleColor.Green;
    Console.WriteLine("You pressed the 'B' key.");
    Console.ForegroundColor = ConsoleColor.Gray;
}
```

Use the start method to start the program.

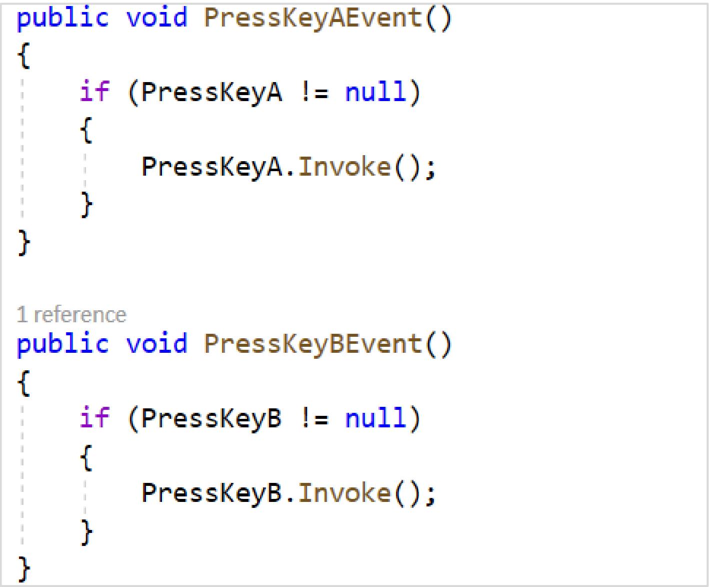
Examples

Input	Output
a	`You pressed the 'A' key.`
b	`You pressed the 'B' key.`
f	`No event handler for key f.`

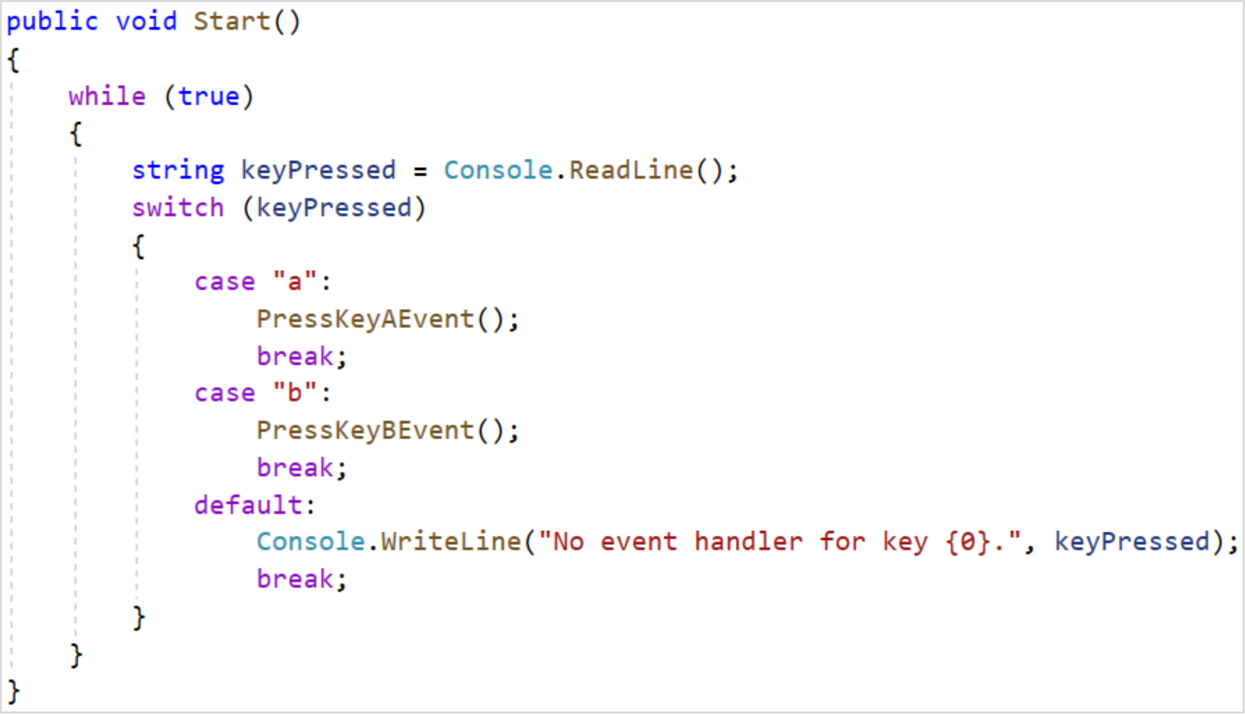
Hints:



```
public event PressKeyEvent PressKeyA = null;
public event PressKeyEvent PressKeyB = null;
```



```
public void PressKeyAEvent()
{
    if (PressKeyA != null)
    {
        PressKeyA.Invoke();
    }
}

1 reference
public void PressKeyBEvent()
{
    if (PressKeyB != null)
    {
        PressKeyB.Invoke();
    }
}
```



```
public void Start()
{
    while (true)
    {
        string keyPressed = Console.ReadLine();
        switch (keyPressed)
        {
            case "a":
                PressKeyAEvent();
                break;
            case "b":
                PressKeyBEvent();
                break;
            default:
                Console.WriteLine("No event handler for key {0}.", keyPressed);
                break;
        }
    }
}
```