BRAINIACS OLYMPIAD

CODING
SAMPLE CHALLENGES

info@brainiacsolympiad.com
www.brainiacsolympiad.com

ORGANIZED BY MINDCSAPE INTERNATIONAL LLC

BRAINIACS OLYMPIAD 2026
CODING SAMPLE CHALLENGES (GRADES 8–9)

Easy challenges - 20

Medium Challenges - 20

Hard Challenges – 10

Easy Challenges

1. Inside Out Wisdom

Joy is welcoming guests into Headquarters. Each time a new person arrives, she shares a message to help them understand their emotions better.

Task:

You will receive one name on a single line. It will not contain spaces or punctuation (even if the proper spelling of the name would include them). Print to the screen the sentence below. The only part of the sentence which should change is the name!

Input

```
Riley
```

Output

```
Your emotions work best when you are the best Riley you can be!
```

Test

```
a)Joy
b)Sadness
c)Anger
```

2. Introduction Program

To find a friend with similar interests, you first need to tell a little about yourself so the program can suggest people with a similar profile.

Task

Write a program that asks for your first and last name, your age, your favorite activity, and your school, and then displays this information in a clear and informative way.

Input:

```
Ariana Grande
21 years old
I love singing
School №1
```

Output:

```
Ariana Grande
Age: 21 years old (born in 2004)
Hobby: I love singing
Studying at: School №1
```

3. Electric charges

You need to calculate the total amount of an electric bill. Every kilowatt-hour (kWh) costs 7.63 cents per kWh.

Task
Write a program that calculates the total amount owed in dollars for a given number of kilowatt-hours used.

Input
The input will be a single integer greater than zero, representing the kilowatt-hours used.

```
1500
```

Output
Print the total amount owed in dollars with up to two decimal places:

```
114.45
```

Additional examples:

Input	Output
900	68,67
1000	76,3

4. Toy Signal Translator

In Andy's room, the toys have a secret system to coordinate their adventures when humans aren't looking. Each toy receives a single-digit signal and performs a special action. Your job is to write a program that translates the number into the toy's action so everyone knows what to do.

Input
A single integer from 0 to 9, representing the toy signal.

```
4
```

Output
Print the action corresponding to the input signal.

```
Hamm hides behind the piggy bank
```

Use the following mapping:

Signal	Toy Action
0	Woody rides Bullseye
1	Buzz flies across the room
2	Jessie spins her lasso
3	Rex roars loudly
4	Hamm hides behind the piggy bank
5	Slinky stretches across the floor
6	Bo Peep signals with her staff
7	Mr. Potato Head winks one eye
8	Aliens raise their hands
9	Bullseye neighs loudly

5. Magic numbers

The wizard Numerus has a sequence of magical numbers. Only positive numbers are useful for his spell. Positive even numbers glow blue, and positive odd numbers glow red. Numerus wants to know how many blue and red numbers are in the sequence.

Task
Write a program that reads a sequence of integers ending with 0 and counts only the positive numbers: how many are even (blue) and how many are odd (red). Negative numbers should be ignored.

Input
The sequence will contain at least 2 numbers and fewer than 15 numbers. The final number `0` indicates

the end of the sequence and should not be counted.

```
-4 5 -33 20 19 0
```

Output

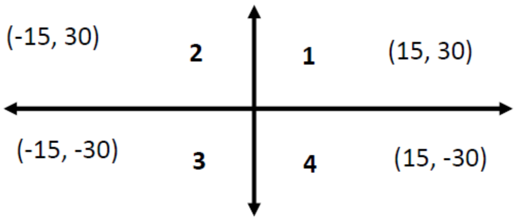
```
Even numbers: 1
Odd numbers: 2
```

Additional examples

```
Input        Output                 Input        Output
2 4 0        Even numbers: 2        -1 -3 0      Even numbers: 0
             Odd numbers: 0                      Odd numbers: 0
```

6. Rocket Navigation

A space probe is flying across a 2D map of a planet. The map is divided into four zones:
Zone 1: X > 0, Y > 0
Zone 2: X < 0, Y > 0
Zone 3: X < 0, Y < 0
Zone 4: X > 0, Y < 0
The probe's instruments provide the coordinates (X, Y) of its current location. Your task is to tell which zone the probe is in.



Task
Write a program that reads the X and Y coordinates of the probe and outputs the zone number.

Input
Prompt for the input as shown below. Enter two integers separated by a space. Neither number can be zero.

```
Enter X and Y separated by a space: 45 90
```

Output
Output the quadrant number (1, 2, 3 or 4) for the point. Use the format shown below.

```
Zone 1
```

Additional examples

Input	Output
`Enter X and Y separated by a space: -15 20`	`Zone 2`
`Enter X and Y separated by a space: -12 -8`	`Zone 3`

7 . Age Scanner of the Nation

A new digital system is being tested to analyze the population of the entire country.
Your mission is to write the Age Scanner, a program that reads the ages of a group of citizens and generates a national statistics report.
You will receive one line of numbers — each number is the age of a citizen.
Your program must determine:
- the youngest citizen

- the oldest citizen
- the average age of the group, rounded down to the nearest integer

Input
A single line with ages separated by spaces.

```
23 45 18 90 34 27
```

Output
Print one sentence exactly in this format:

```
Youngest citizen: 18, Oldest citizen: 90, Average age: 39
```

8. **Number Frequency**
In this problem you are given a list of positive integer values between 1 and 100 inclusive. Determine how many times each value is entered. Display each number entered and how many times it was entered.

Input
Prompt the user as shown in the sample input below. Then enter some number of positive integers between 1 and 100. Any number of values can be entered. Each will be separated by a space. At least one non-zero number will be entered. The last integer in the list will be a zero indicating the end of the data. The zero is NOT part of the list. Input
Enter your numbers separated by a space

```
44 44 78 44 100 1 1 1 100 44 55 0
```

Output
A listing of each number entered and how many times that number was entered. The output must be in order from the lowest number entered to the highest number entered. You must correctly use the words “times” and “time” as shown in the sample output below. Display the output on the monitor. See the sample output below.

```
1 occurs 3 times
44 occurs 4 times
55 occurs 1 time
78 occurs 1 time
100 occurs 2 times
```

9. Wiping secret files
A spy needs to permanently erase two secret files. Each file has a certain size in bytes, and the spy’s erasing device can overwrite data at a given speed in bytes per second.

Task
Write a program that calculates the total time (in seconds) required to completely erase both files. The program should:
Read four positive integers from the console: size1 speed1 size2 speed2.
Compute the time for each file as size ÷ speed.
Output the sum of both times as a single integer.

Input

```
3000 1000 12000 4000
```

Output

```
6
```

Additional examples

Input	Output
`2048 1024 500000 10000`	`52`
`5000 500 3000 1000`	`13`

10. Triangle Types

Sipho needs your help distinguishing between different types of triangles. Given three angles, given as three positive integers in degrees, you must first determine if they constitute the angles of a triangle. Note that the angles in a triangle must add up to 180°. If the angles form a triangle, you must then determine if it is an equilateral, isosceles, or scalene triangle.
Write a program that, when given three angles, determines what type of triangle they make. If no triangle can be made with the given angles, output "IMPOSSIBLE". Otherwise, if it makes an equilateral, isosceles, or scalene triangle, output "EQUILATERAL", "ISOSCELES", or "SCALENE" respectively (without quotation marks).

Note: An equilateral triangle has all three angles equal, an isosceles triangle has only two angles equal and a scalene triangle has three different angles.

Examples:
Input: 30 60 70 Output:IMPOSSIBLE
Input: 40 40 100 Output:ISOSCELES
Input: 50 60 70 Output:SCALENE

Test your program with the following cases:
a) 60 60 60

b) 80 20 80

c) 75 34 12

d) 79 23 78

11. Radio towers

The city has several radio towers of various heights. A pair of radio towers can communicate with each other if their difference in height is less than or equal to 10 m.
Write a program that receives the number of radio towers and the height of each of the radio towers as input and outputs the number of radio tower pairs that can communicate with each other.

Examples:
Input: 3
1 11 15
Answer: 2
Input: 5
8 29 22 16 30
Answer: 5

Tests your program with the following cases:
Give your answer as a number only with no spaces, e.g. 1234
3a) 2
19 21
3b) 5
1 2 3 4 1
3c) 10
21 43 23 45 28 31 49 13 54 32
3d) 34
5 25 38 6 2 29 33 50 3 29 19 18 9 18 13 22 21 30 41 19 16 40 41 7 35 1 41 22 25 1 40 9 3 46

12. Cellphone

A cellphone service provider has offered a new subscription plan. The specific pricing for daytime, evening and weekend usage is given below:

Daytime - 100 free minutes then 80c per minute
Evening - 70c per minute
Weekends - 50c per minute
Your task is to write a program to determine how much the new subscription plan will cost, given how many minutes are used for daytime, evening and weekends. These are given as three non-negative integers in the order of daytime, evening, then weekend. Your program should output the total price in cents the subscription plan costs, as shown in the examples given below.

Examples:
Input: 45 0 1 Output: 50c

Input: 162 61 66 Output:12530c

Input: 82 89 15 Output:6980c

In the first example given, 45 daytime minutes, 0 evening minutes and 1 weekend minute were used. As the first 100 daytime minutes are free, no cost is allocated for daytime minutes. As no evening minutes were used, no cost is allocated for the evening, and as only 1 minute was used on the weekend, the total cost comes to 50c.

13. Determinant

Kevin, a university student, is struggling with his mathematics homework and can't find his calculator. Help him find the correct answers for the following mathematical operation.
Given as input 4 space-separated integers a b c d, write a program to output the value ad – bc. The value of each integer is between -10000 and 10000.
Example: 4 5 2 3
Answer: 2 (4*3 – 5*2)
Test your program with the following cases:
Give your answer as a number only with no spaces, e.g. 1234
1a) 4 -2 6 3
1b) 6 7 7 7
1c) 78 21 43 65
1d) 9999 -400 173 77

14. Hidden Signal Locator

In a data sequence, a single character signal is hidden among other characters.
The goal is to determine the **position** of this special character within the sequence.

Input

Line 1: A single character C — the signal to locate.

Line 2: A string S of length 10 to 1000, containing exactly one occurrence of C.
AAAAAAAAZAAAAAAAA

Output
Print the zero-based index of the signal in the format:

Z found at index 8

15. Ruler Helper

You have a ruler that measures lengths in centimeters, but you need the measurement in meters and kilometers. Write a program that converts the given length from centimeters to meters and kilometers.

Input

A single integer L – the length in centimeters.

```
123456
```

Output

Print the length in meters and kilometers, separated by a space, rounded to exactly two decimal places.
1 meter = 100 cm
1 kilometer = 100,000 cm

```
1234.56  1.23
```

16. Building pyramids

The weather has kept everyone inside. The math teacher has given everyone an interesting assignment. Take cubes of sugar and build pyramids. Then she asks how many cubes you used. The problem is each team built a pyramid of a different height. Can you help everyone count their cubes quickly?

Input

Read 1 positive integer, the height of the pyramid. Each layer is a full square and has one less cube along its length than the layer beneath it. The top layer is a single cube. The maximum value will be 1000 (it has been a long winter.)

```
6
```

Output

Print the number of cubes in the entire pyramid.

```
91
```

Explanation

The bottom layer has 6x6 cubes (36). The next higher layers have 25, 16, 9, 4, and 1 cubes each, for a total of 91.

17. Solar Panel Efficiency

A research team is testing new solar panels in different weather conditions.
Each panel produces a certain amount of energy (in watts). But due to dust, clouds, and temperature changes, some readings are negative — these represent faulty measurements and must be ignored.

Task

Write a program that reads a sequence of integer watt values.
Stop reading when the value -999 is encountered (this end value should not be counted).
Only positive numbers represent valid energy readings.
Your program must calculate:
- the total energy (sum of all positive readings)
- the number of valid readings
- the average output, rounded down to the nearest whole number (If no valid readings, print 0.)

Input

```
50 -3 20 -10 40 -999
```

Output

```
Total energy: 110
Valid readings: 3
Average output: 36
```

18. Alien Treasure Pods

An alien spaceship has a long corridor with N glowing pods placed in a straight line.
The pods light up in alternating colors:
First pod is GREEN
Second is PURPLE
Third is GREEN
Fourth is PURPLE
... and so on.
Energy crystals are stored in these pods:
Pod 1 contains 1 crystal
Pod 2 contains 2 crystals
Pod 3 contains 4 crystals
Pod 4 contains 8 crystals
And so on — the number of crystals in each pod doubles every time.

Task
Write a program to calculate the total number of crystals stored in all GREEN pods.

Input
A single integer N — the total number of pods.

```
3
```

Output
A single integer — the total number of crystals inside all green-colored pods.

```
5
```

Explanation: Green pods: pod 1 (1 crystal), pod 3 (4 crystals)

19. Prime Numbers

Prime numbers form a fundamental part of mathematics. Recall that a prime number is a positive integer greater than 1 and which is divisible only by itself and 1. The first five prime numbers are 2, 3, 5, 7 and 11.

Given a positive integer ��, write a program which outputs the ��-th prime number. n
Examples:
Input: 3 Output: 5
Input: 10 Output: 29
Input: 15 Output: 47

Test your program with the following cases: Give your answer as a number only with no spaces, e.g. 1234
a) 30
b) 101
c) 12321

20. Robot Step Counter

A delivery robot moves inside a warehouse on a straight path.
Each movement command is either:
F – move forward 1 meter

B – move backward 1 meter
Your job is to determine the robot's final position relative to its starting point.

Task
Write a program that reads a string consisting of the letters F and B only.
Calculate the final displacement:
Forward steps count as +1
Backward steps count as –1
Then print the final position.

Input
FBFFBFBF

Output
A single integer representing the net position.
2

Additional examples

Input	Output
FFFFBFB	3
BBBBFF	-2
FBFBFBFB	0

Medium challenges

1. Your Shift Is Over

Given a list of integers, "shift" the list a set number of positions to the left. For example, given the list:
3 6 18 -7 99 12
and a "shift" of 2, the result would be:
18 -7 99 12 3 6
The result makes it appear each number has been shifted 2 positions to the left. Numbers that shifted off the left edge (3 and 5 in the example) resurface in their proper location on the right end of the list.

Input
A list of valid integers. There will be at least 1 integer entered and not more than 100. The number 0 will indicate the end of the list. The 0 must not be considered part of the list. The shift number is also entered. This number will be greater than 0 and less than or equal to the number of integers in the list. Read all the data from the console.
Enter the shift number

```
4
```

Enter the numbers in list separated by space. End with 0.

```
4 5 6 7 8 0
```

Output

The modified list shifted N positions to the left. N represents the shift number as described above.

```
8 4 5 6 7
```

Additional examples

Input	Output
3 1 2 3 4 5 0	4 5 1 2 3

2. Weighted Sum

The goal in this problem is to sum all the digits in a number. However, each number must be “weighted” by its relative position in the number. For example, given the number 57 the weighted sum is 19. The weighted sum is calculated as: (5 * 1) + (7 * 2).

Given the number 482178, the weighted sum is 113. The weighted sum is calculated as:

(4 * 1) + (8 * 2) + (2 * 3) + (1 * 4) + (7 * 5) + (8 * 6).

Input

Enter a positive integer number. Read the number from the console.

```
Enter the number:
57
```

Output

Display the weighted sum of the number.

```
Weighted sum is 19
```

Additional examples

Input	Output
4056	59
2468	60

3. ALPHABET SQUARE

Assign each letter of the alphabet (upper or lower case) a number according to its ranking: A = a = 1, B = b = 2, C = c = 3, D = d = 4, E = e = 5, etc. Given a single letter as input, print a square consisting of the given letter with side length equal to the ranking of the letter, as in the examples.

Input: D

Output:

```
D D D D
D D D D
D D D D
D D D D
```

Input: e

Output:

```
e e e e e
e e e e e
e e e e e
e e e e e
e e e e e
```

Use the following values to test your program

a: C

b: g

c: T

4. Checkered Board

Chess, Checkers and many other games are played on an 8 by 8 checkered board. Many other games can be played on boards of different sizes.

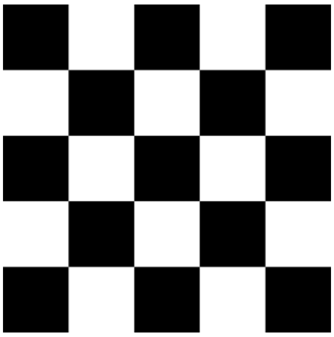


Figure 1. 5 by 5 checkered board

Task

Write a program that reads in the checkered board's side length S as input. Your program must output the number of black squares on a S by S checkered board. Assume the top right square of the board is always black.

Examples Input: 5
Output 13
Input: 8
Output: 32

Test your program with the following cases:

1.a	10
1.b.	21
1.c.	36
1.d.	111

5. Tree Growth

In a virtual forest, a tree’s name determines its growth speed. Each letter in the tree’s name is assigned a value as follows:

- Vowels (a, e, i, o, u) are 5 points each.
- Special consonants (b, c, d, f, g) are 3 points each.
- All other consonants are 2 points each.

The sum of these points determines the tree’s growth speed as follows:

Sum Points	**Growth speed**
Less than 25	Slow
25 to 35	Medium
Greater than 35	Fast

Task

Write a program which reads in the name of the tree and outputs the tree’s growth speed.

Examples:
Input: baobab

Output: Slow
Explanation: The sum points of baobab is 3 + 5 + 5 + 3 + 5 + 3 = 24. Since 24 is less than 25 the growth speed is Slow.

Input: pineapple
Output: Medium
Explanation: The sum points of pineapple is 2 + 5 + 2 + 5 + 5 + 2 + 2 + 2 + 5 = 30. Since 30 is in the range 25 to 35 the growth speed is Medium.

Test your program with the following cases:
2.a. oak
2.b. magnificent
2.c. sequoia
2.d. yellowwood

6. Ordered squares
A number is an ordered number when its digits are in an increasing sequence. For example 136 is an ordered number but 163 is not, 225 is an ordered number, but 252 is not. A single digit number is always ordered.
A square is a number obtained by multiplying a positive whole number by itself. 16 is a square (4X4)
Write a program that will print out all the ordered squares up to and including a given value (V).
Separate the ordered squares by commas or spaces.
V will be a positive whole number less than 100 000
Input: V = 30
Output: `1, 4, 9, 16, 25`

`Additional exmaples:`
(a) V = 100
(b) V = 1 000
(c) V = 2 000

7. Telephone Directory
In a telephone directory the names of streets and suburbs are shortened by removing all vowels (a, e, i, o, u) unless the vowel appears at the beginning of a word, in which case it is kept.
Write a program that will automatically shorten any address in this manner. Punctuation and numbers must be reproduced as in the input.

Example 1: Input: Woodside
Output: Wdsd
Example 2:
Input: Observatory
Output: Obsrvtry

Record the results of the following test cases:
a. Johannesburg
b. Apple Street, Claremont
c. 21 Cavendish Avenue, eThekwini

8. The beach that makes you old
So, there's this beach, and it makes you old.
And there's a group of high school students who want to visit the beach that makes you old. They want to rent a car to go on a trip, but most car rental places will not rent to you unless you are at least 25.

So, they want to go to the beach that makes you old. The beach that makes you old doesn't make sense logically. Maybe it's the sand, or the rocks, or the coral. We don't know.

For every hour a person remains at the beach, their body ages one month. Write a program to calculate how long students should remain at the beach, so that they can rent a car.

Plot Twist After the students return from the beach, their driver's licenses still show their birth dates. Checkmate, students.

Input
The first line is the number of students. Each following line gives a student's name and their age in years and months. All ages are less than 25 years.

```
5
Asok 16 3
Brenda 17 11
Cariana 15 9
Due 17 0
Ezhno 15 6
```

Output
In the same order as the input, print each student's name, one space, and the number of hours they must spend at the beach to reach age 25.

```
Asok 105
Brenda 85
Cariana 111
Due 96
Ezhno 114
```

9. Security Digit
To check whether an account number is genuinely a number allocated by the bank, the Pomme Bank of Paris uses the following technique. All the non-zero digits in the number are multiplied by each other. All the non-zero digits of the resulting number are again multiplied by each other – and so on until a single digit is left. That is the security digit.
Write a program that will provide the security digit for any number with up to 20 digits.

Example 1:
Input: Number? 469795
Output: Security digit = 8

Example 2:
Input: Number?1234239003
Output: Security digit = 9

Test the following cases:
a. 193
b. 901090506
c. 1567890123

10. Justifying Text

Write a program that will take the provided text and print it out in a column of a specified width (just like in a newspaper). Words at the end of a line that would end beyond the length specified, must be moved to the next line. Spaces must be added between the words so that each line is exactly the specified width and the spaces are as evenly distributed over the line as possible. If only one word fits in a line, that word must be left-aligned. The width of a column will always be greater than or equal to nine.

Use the following text: “A great discovery solves a great problem but there is a grain of discovery in the solution of any problem”

Example 1:
Input: 20

```
A great discovery
solves a great
problem but there is
a grain of discovery
in the solution of
any problem
```

Example 2
Input: 25

```
A great discovery solves
a great problem but there
is a grain of discovery
in the solution of any
problem
```

Test the following cases:
a. Width? 35
b. Width? 60
c. Width? 10

11. Order Letters

Write a program which, given a string consisting of uppercase letters (A, B, ..., Z) separated by + characters, will output that string with the letters in reverse alphabetical order.

Input
A single string consisting of uppercase letters separated by + characters.

Output
The string with the letters in reverse alphabetical order: All Z’s before all Y’s, and all Y’s before all X’s, and so on. The + characters must not move.

Examples:

Input	Output
Q+B+Q+Z	Z+Q+Q+B
B+B+U+U+P+G+S+B+H+D	U+U+S+P+H+G+D+B+B+B
B+B+A+C+B+C	C+C+B+B+B+A

12. Robot Speech Analyzer

Your robot assistant, R-42, is learning to speak like a human. To measure its progress, scientists analyze the average length of real words in everything the robot says.
However:
Only alphabet letters (A–Z, a–z) count as part of words.
Any symbols, numbers, emojis, punctuation, or sound effects must be ignored.

Consecutive letters form a word.
Your job is to compute the average length of all valid words, rounded to two decimal places.

Task
Write a program that analyzes the robot's message and calculates:
- Average number of letters per word, using only the English alphabet.
- Round the result to two decimal places.

Input

```
Beep! System reboot complete.
```

Output

```
6.00
```

Words considered: Beep (4), System (6), reboot (6), complete (8)
Average = (4 + 6 + 6 + 8) / 4 = **6.00**

13. Rock Paper Scissors Lizard Spock
Implement the game: Rock Paper Scissors Lizard Spock
The Rules Are:
Scissors cuts Paper Paper covers Rock Rock crushes Lizard Lizard poisons Spock Spock smashes Scissors
Scissors decapitates Lizard Lizard eats Paper
Paper disproves Spock Spock vaporizes Rock Rock crushes Scissors
You will receive the choices from both players at the same time in the form of:
Choice1 Choice2

You are guaranteed to only receive one game choice at a time. You are guaranteed that the game choice will be formatted properly, and will exist.

If the same choice is made for both sides, the result is a tie. State the winner in terms of the first choice made (the choice to the left of the space). Else, state the outcome and the winner.

Input
Scissors Lizard

Output
SCISSORS WINS, Scissors decapitates Lizard

Input
Scissors Rock

Output
SCISSORS LOSES, Rock crushes Scissors

Input
Rock Rock

Output
TIE, ROCK does not affect ROCK

14. Happy Number Checker
Some numbers are called happy numbers. A happy number is a number that eventually reaches 1 when you repeatedly replace the number with the sum of the squares of its digits. Numbers that never reach 1 and start repeating are called unhappy numbers.

Task

Write a program that determines whether a given positive integer is a happy number or not.
Start with a number N > 0.
Generate the next number in the sequence by summing the squares of its digits.
Repeat this process until the number becomes 1 (happy) or the sequence repeats (unhappy).
Print Happy Number if the sequence reaches 1, or Not a Happy Number if it repeats without reaching 1.

Example 1
Original input: 49
Sequence generated:
49 -> 97 -> 130 -> 10 -> 1
Output: Glad Number
The sequence was generated as follows:
$4^2 + 9^2 = 97$
$9^2 + 7^2 = 130$
$1^2 + 3^2 + 0^2 = 10$
$1^2 + 0^2 = 1$

Example 2
Original input: 11
Output: Not a Glad Number
The sequence was generated as follows:
$1^2 + 1^2 = 2$
$2^2 = 4$
$4^2 = 16$
$1^2 + 6^2 = 37$
$3^2 + 7^2 = 58$
$5^2 + 8^2 = 89$
$8^2 + 9^2 = 145$
$1^2 + 4^2 + 5^2 = 42$
$4^2 + 2^2 = 20$
$2^2 + 0^2 = 4$
Sequence will repeat and not reach 1

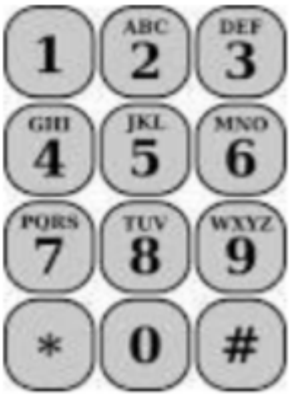
Input
Prompt for the input as shown below. Enter an integer number > 0. You do not need to edit this data. Data will be entered from console. You do not need to edit the input data. Rerun your application to test each test case.

Output

Happy Number or Not a Happy Number.

15. Let’s Make a Call

Given an alphanumeric phone number, convert it to all digits. For example, given 1-800-FLOWERS, your application will convert it to 1-800-3569377. Notice the hyphens maintain their location to match their position in the original phone number. An image of a phone keypad is provided below. All letters in the phone number will be uppercase only. Run your application once for each set of test data.



Input
A valid alphanumeric phone number consisting of the number 1 followed by a dash followed by any combination of the digits 1 through 9, dashes, and uppercase letters in the alphabet. Read the phone number from the console.
Enter the phone number:
1-800-FLOWERS

Output
The equivalent phone number converted to all numbers.
1-800-3569377

16. Island Jumper

You are a daring adventurer exploring a chain of mysterious islands. Each island has a magical number carved into it, which tells you how many islands you can jump forward from there. Your mission is to start on the first island and reach the treasure on the last island without overshooting it.

Rules:

- The first number in the list tells you **how many islands** there are (not a jump).
- Each following number represents the **jump distance** from that island.
- You can only jump **forward**, never backward.
- Landing exactly on the last island means you reach the treasure — print Treasure Found!
- If you land past the last island or get stuck on an island with 0 jump length, you fail — print Stranded!

Input
The first number N is the number of islands.
The next N numbers are the jump lengths from each island.
6 3 2 1 0 2 3

Output
Stranded!

Additional examples
Input
5 3 1 0 1 3

Output
Treasure Found!

17. Equivalent Strings

In this problem you will determine if two strings are "equivalent". Equivalent means one string can be transformed into the other by substituting one letter for another. For example, consider the two strings “cocoon” and “zyzyyx”. These would be considered equivalent because: c-> z, o -> y, n -> x. Each letter in one string maps to one and only one letter in the second string and vice versa.

On the other hand, the two strings “banana” and “orange” are not equivalent. This is because ‘a’ in banana would have to map to ‘r’, ‘n’, and ‘e’ in orange. There is no one to one substitution that works. Based on the definition of “equivalent” for this problem, strings of different lengths cannot be equivalent.

Input
The first string will be entered on line 1. The second string will be entered on line 2. Each string will have one or more lowercase letters. The strings will not necessarily be the same length. Read the input from the console.
Enter first string
zyzyyx
Enter second string
cocoon

Output
Display “equivalent” if the strings are equivalent based on the description above. Display “not equivalent“ if the strings are not equivalent. Display the output on the monitor. See the sample output below.
Equivalent

18. DNA Corruption

Bongi, the astronaut, needs your help. Without the protection of the atmosphere, her DNA is being corrupted by the sun’s radiation! Luckily, our cells come with backup DNA. Help Bongi figure out which strands of her DNA have been corrupted.
A strand of DNA is a string of the 4 characters: T, A, C and G. The characters form two pairs. The first pair is (A, T). The second pair is (G, C). If two characters are in the same pair, they are considered opposite. That is, T is the opposite of A, and A is the opposite of T. G is the opposite of C, and C is the opposite of G.

Task
Write a program that takes two strings of length 5 as input. The first string is the original DNA, and the second string is the backup DNA. Your program must output “OK” if every character in the backup strand of DNA is the opposite of the corresponding character in the original strand of DNA. Otherwise, output “CORRUPTED”.

Example 1:
Input: ACTAG TGATC
Answer: OK
Explanation: Reading from left to right of the original DNA: The opposite of A is T. The opposite of C is G. The opposite of T is A. The opposite of A is T. The opposite of G is C. All the characters in the backup are the opposite of the original. The DNA is “OK”.

Example 2:
Input: GTACG CACGC
Answer: CORRUPTED
Explanation: The third character of the original DNA is
A. The opposite of A is T, but the third character of the backup DNA is C. The DNA is "CORRUPTED".

Test your program with the following cases:
a) ACACA TGTCT

b) ATGGC GCCAT
c) TCGCT AGCGA
d) TTGAC TTGAC

19. Backward Echo

At school, Thabo is playing a challenge game with his English teacher. The teacher writes one lowercase letter (a–z) on the board.
Thabo must then echo back a sequence of letters by moving backwards through the alphabet, skipping one letter each time.

For example:
If the teacher writes e, Thabo says: e c a
If the teacher writes d, Thabo says: d b
Thabo stops when he reaches before 'a'.

Task:
Write a program that reads a single lowercase letter and outputs the sequence of letters Thabo echoes.

Input:
A single character between a and z.
g

Output:
The letters moving backwards, skipping one each time.
geca

Try your program with these test cases:
a) q
b) k
c) a
d) z

20. Reading Level

A simple way of checking the reading level of a given piece of text is to determine the average length of the words in the text.

Task: Write a program that will calculate the average length of the words for any given piece of text and give the answer rounded off to two decimal places. Only the 26 letters of the alphabet (upper and lower case) must be taken into consideration. The given text may include other punctuation marks which should NOT be considered part of the text.

Examples:
Input: How now brown cow?
Output:3.50

Input: "Stay inside the house", she said.
Output:4.17

Test your program with the following cases:
a) Mary had a little lamb
b) Upon this followed irritation, aggravation and frustration.
c) Most newspapers – and all that follow AP style – insert a space before and after the em dash.
d) "Stop!", shouted Chris. He said to Emily, "Why were you running in the hall?"

Hard challenges

1. Delta Inversion

Given a sequence of integers, write a program to 1) compute the deltas (difference between values), 2) invert the deltas, and 3) compute new sequence based on the new deltas. For example, if you started with the following sequence:
33 47 48 19 86 6
The deltas would be:
14 1 -29 67 -80
The inverted delta values would be:
-14 -1 29 -67 80
Finally, applying these inverted deltas and starting with the first number of the original sequence gives us this new sequence:
33 19 18 47 -20 60

Input
The first line of input indicates the number of integer sequence (up to 3). The first integer on each subsequent line indicates the number of values in a sequence(up to 32) and is followed by the integer sequence. The integer values in a sequence may be between -999 and 999 (inclusive).

```
2
7  -321  524  12  0  924  -658  -2
5  246  -651  -650  -650  31
```

Output
The program must print the new integer sequences, one per line.

```
-321 -1166 -654 -642 -1566 16 -640
246 1143 1142 1142 461
```

2. Ciphers

In this problem, each letter of the alphabet corresponds to a number using the scheme: a=1, b=2, c=3, ... y=25, z=26. To encode a message, an encryption key word is added to the message. The key word is the first word in the message that is five or more characters long. For example, if the message were: "give me liberty or give me death," the key word would be "liberty." The encrypted message would be: s r x j e y k u k g w l s n d p k a w g d p n c y z
The first letter, s, results from adding 7 for the letter "g" to 12 for the letter "l." The sum is 19, which creates the letter "s." The other letters are calculated in a like manner. The algorithm you develop should wrap around to the beginning of the alphabet if the sum of two letters exceeds 26. For example, when the letter "l" (12) is added to "y" (25), the sum is 37. Since there is no 37th letter in the alphabet, you should subtract 37 from 26 to get the number 11, which equates to the letter "k." The key word is repeated indefinitely to the end of the message.

```
give me liberty or give me death
libe rt ylibert yl iber ty liber
--------------------------------
srxj ey kukgwls nd pkaw gd pncyz
```

(Spaces are left between the words in this example for clarity. Do not include spaces in your results.)
Every message will have at least one five-letter word.
Spaces are not encrypted and not included in the output. Only messages in lowercase letters will be entered. No punctuation or special characters will be entered.

Input

Consist of a sentence or phrase entered via the standard input device and the key.

```
a stitch in time saves nine
stitch
```

Output

```
t m c c w k a c w n l u x m j p h a g c w y
```

Additional examples

Input

```
it is a rainy day
rainy
```

Output

```
a u r g z j b r b x v b h
```

Input

```
it only hurts when i laugh
hurts
```

Output

```
q o g h e g c m l m a r z y g q g s o z p
```

3. TRANSFORMATION

Some doors can be unlocked using a key card – a plastic card with holes in it. A key card may be square like this: (O means a hole, X means no hole); but it may also be a rectangle or a triangle.

```
XXXO
XXOX
XOXX
XXOX
```

However in order to work, these specific key cards have to be rotated or flipped according to one or more of these transformation instructions:

R = Rotate 90° to the right

T = Top to bottom flip: you see the back – upside down

L = Left to right flip: you see the back – mirrored

Task:

Write a program that will take the given key card, rotate, and/or flip it as instructed, and print out the resulting key card. No card will be more than 40 by 40 spaces.

Example:

Input:

```
XXXO
XXOX
XOXX
XXOX
Transformation: R, L
```

Output:

```
XXXX
XXOX
XOXO
OXXX
```

Explanation:
After rotating by 90°

```
XXXX
XOXX
OXOX
XXXO
```

After a left to right flip:

```
XXXX
XXOX
XOXO
OXXX
```

Test your program with

```
(a) XXXO
    XXOX
    XOXX
    XXOX
```

Transformation: T

```
(b) XXXO
    XXOX
    XOXX
    XXOX
    XXXX
```

Transformation: T, L, R

```
(c)    O
      XXO
     OXXXO
    XOOOXXX
```

Transformation: R, T, L

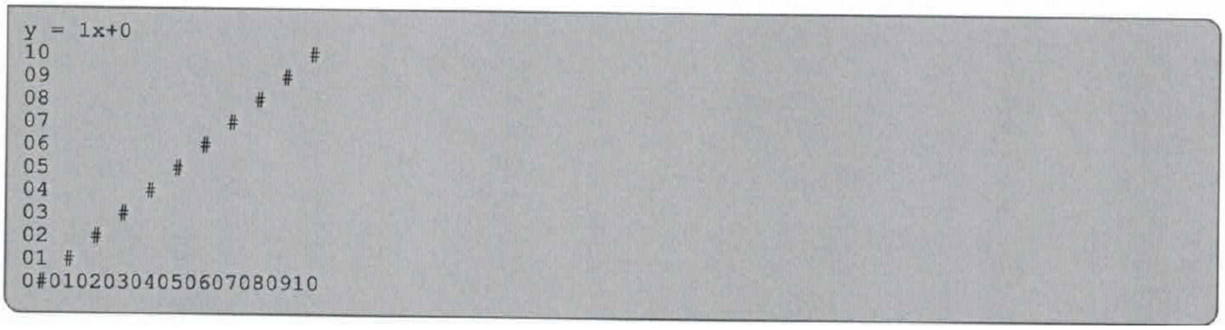
4. y = mx+b
Monty Burns is fed up with his army of sycophants who tell him whatever it is they think he wants to hear. He just paid for an independent audit of his finances, and things aren't looking good. You seem like you know your way around the ol' teletype-a-graph, and Monty wants you to help!
Help Monty chart his finances so he can figure out if he can afford that second gold-plated back scratcher or not.

Input
Given integers m, b, and the size of your square output grid all separated by spaces on a single input line, graph the output. Note, m can be negative! Plan accordingly.

```
1 0 10
```

Output
Write the equation for the line in slope-intercept (y=mx+b) format, then graph it (only in quadrant 1, and only for the square size listed in your input). Your graph should show the intersection with the y-axis by having the second number on the y-axis replaced with a point (#). Other than (0,0), do not show intersections with the X-axis (stop before intersecting the X-axis). Spacing for your X and Y axis labels should always be 2 characters wide, left padded with zeros if needed. All points on the graph should be on the same horizontal line as their Y coordinate. For their X coordinate, place the point (#)above the second number in the x-axis labels (e.g. above the "9" in "09", above the "3" in "13", etc.). Do not extend

the graph line past the boundary of the graph size, even if one of the coordinates would have been within size -- both must be within size to show the point.

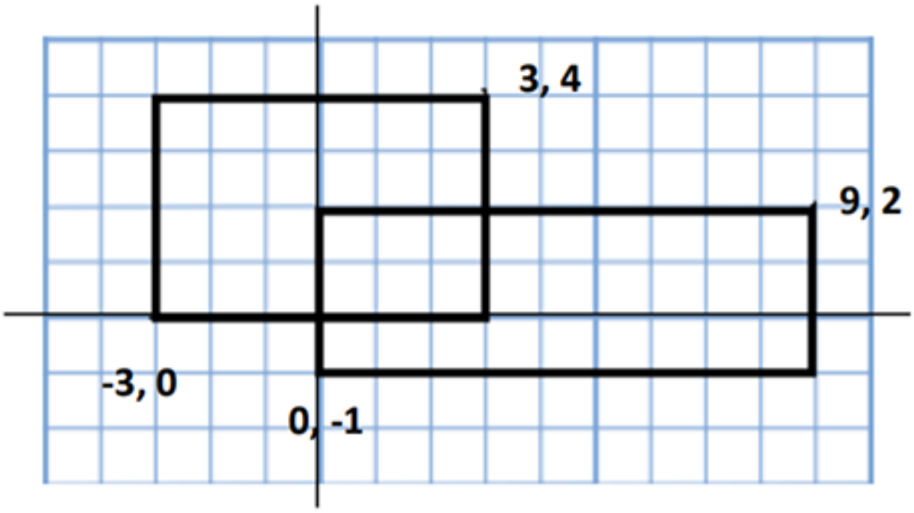


```
y = 1x+0
10                   #
09                 #
08               #
07             #
06           #
05         #
04       #
03     #
02   #
01 #
0#01020304050607080910
```

Special Case:
In the case where slope is zero, that is the equation for a horizontal line. In the interest of simplicity, you can simply write out the equation as you always have substituting zero for "m", however, your line must be a horizontal line on that Y-intercept line -- following the rest of the rules as outlined above.

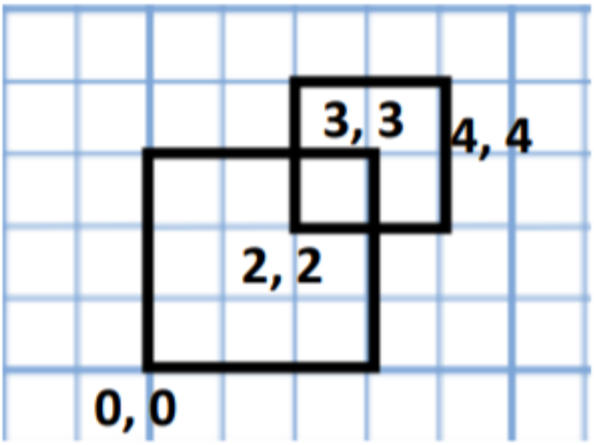
5. Overlapping Rectangles

Given two rectangles that overlap, determine the total area of the two rectangles and the shared area of the two rectangles. A rectangle can be defined by the x and y coordinates of its lower left corner and its upper right corner. In this problem you will be given the coordinates to define two rectangles. The rectangles will intersect (overlap) in some way. For example, given the rectangles defined by rectangle 1 (-3, -0) and (3, 4) and rectangle 2 (0, -1) and (9, 2).



The area of a rectangle equals its width times its length. The area of the rectangle on the left is 24 (6 times 4). The area of the rectangle on the right is 27 (9 times 3). The area of the overlap section is 6. This means the total area of the rectangles is 45 (24 + 27 – 6).

This second example uses the rectangles defined by rectangle 1 (0, 0) and (3, 3) and rectangle 2 (2, 2) and (4, 4).



The area of the rectangle on the left is 9 (3 times 3). The area of the rectangle on the right is 4 (2 times 2). The area of the overlap section is 1. This means the total area of the rectangles is 12 (9 + 4 – 1). The rectangles will always overlap in some way. The rectangles will never be the same rectangle.

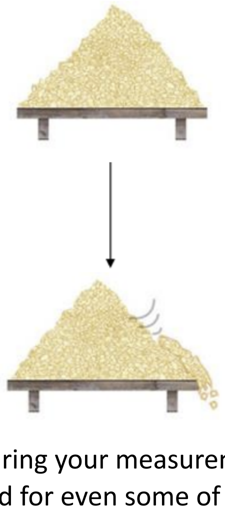
Input
The first line of input will be four integer values each separated by a space. The second line of input will be four integer values each separated by a space. Read each number from the console.

```
Enter corners of rectangle 1:
-3 0 3 4
Enter corners of rectangle 2:
0 -1 9 2
```

Output
Display the area where the rectangles overlap. Display the total area of the two rectangles. Display the output on the monitor. See the sample output below.

```
overlap area 6.0
total area 45.0
```

6. Sandpile
In front of you is a large sandpile on a wooden table. The wooden table has length L. You have measured that i units from the left the sandpile has height hi.



During your measurements, you accidentally bumped the table causing your sandpile to change shape and for even some of the sand to fall on the floor!
The sandpile changes shape according to the following avalanche rule:

If a height of hi is more than a unit taller than either hi+1 or hi-1, the height hi avalanches. Meaning that it sends one unit of sand to the left and one unit of sand to the right. Note an avalanche can trigger another avalanche, and that the edge of the table h0 and hL+1 have height 0. When the height h1 or hL avalanches, one of the units of sand falls off the table.
Making use of the avalanche rule, write a program that will be able to tell how much sand fell off the table. (Hint: The tests guarantee that no matter in which order the avalanches occur, the amount of sand that falls off the table is the same.)

Input
The first line of input is L the length of the table. The second line of input are the heights h1, h2, …, hL.
Output
The amount of sand that falls off the table.

Example 1
3
1 4 1
Answer: 2

Example 2
3
3 2 1
Answer: 3

Example 3
4
1 2 3 1
Answer: 2

Explanation of first example: The sandpile goes through the following avalanches highlighted in red. Sand falls off the table during the second and third avalanches.

1 4 1 → 2 2 2 → 0 3 2 → 0 4 0 → 1 2 1

Tests
a) 3
3 1 2
b) 5
1 1 6 1 1
c) 4
4 3 2 1
d) 5
1 7 1 7 1

7. Digit Word

A digit word is a word where, after possibly removing some letters, you are left with one of the single digits: “ONE”, “TWO”, “THREE”, “FOUR”, “FIVE”, “SIX”, “SEVEN”, “EIGHT” or “NINE”. For example, the word “BOUNCE” is a digit word as it becomes “ONE” after removing the letters ‘B’, ‘U’, and ‘C’. The word “ENCODE” contains the letters ‘O’, ‘N’, and ‘E’, but not in the correct order, therefore it is not a digit word.

Write a program that, when given a word W containing only uppercase letters of the alphabet, determines if it is a digit word. If W is not a digit word, output the single word “NONE”. Otherwise, output the English name of the lowest digit which W contains in uppercase. For example, if W contains both “ONE” and “EIGHT”, then output “ONE” (without quotation marks).

Examples:
Input: BOUNCE Output:ONE
Input: ENCODE Output:NONE
Input: SPHINX Output:SIX
Input: FOUNDER Output:ONE
(note that FOUNDER also contains FOUR, but ONE is less than FOUR)
Test your program with the following cases:
a) TWOSEVENFIVE
b) TELECOMMUNICATIONS
c) ZUEXNIMTDJFDJIHSBDRSJDENNBXCSE
d) XEULVFMAATXXZOIFRZCMUQSGHAVXCJUJ HXAXOPVZJTD
e) ZLXQMVBEHHEPTJDNIBCREPLLGUSJCPXA NBJIQYJYJPCYGAVWBPNYGE

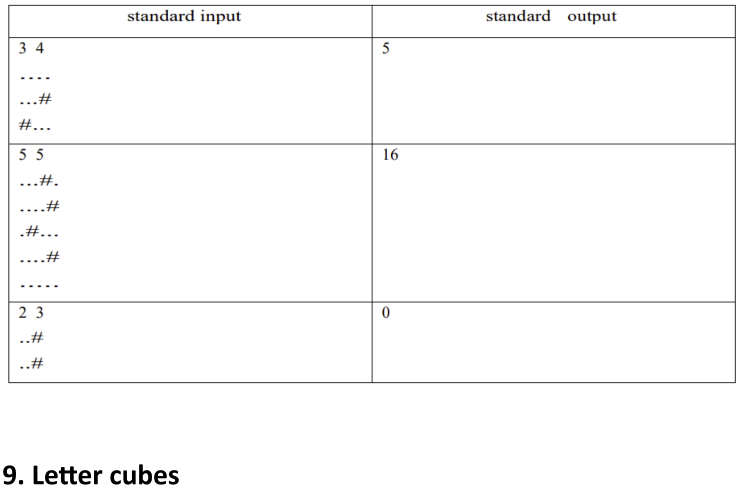
8. Grid Walk

You find yourself at the top-left cell of an N x M grid and would like to make your way to the bottom-right cell, through a sequence of steps. For each step, you are allowed to move either one cell to the right, or one cell down. However, you are not allowed to move into a “blocked” cell.

Write a program to determine the number of ways there are of reaching the bottom-right cell. Note, there might be 0 such ways of reaching the bottom-right cell. It is guaranteed that the top-left cell is not blocked.

Input
The first line contains two space-separated integers, N and M (1 ≤ N, M ≤ 1000). The next N lines contain M characters each, describing the rows of the grid. A ‘.’ represents an empty cell, while a ‘#’ represents a blocked cell.

Output
Output a single integer, the number of ways of reaching the bottom right cell of the grid. This number can be extremely large, so output only the remainder when divided by 108 . For example, if the answer was 1038000432, you would output 38000432. If the answer was 1700060120, you would output 60120.



standard input	standard output
3 4 ....# #...	5
5 5 ...#. ....# .#... ....#	16
2 3 ..# ..#	0

9. Letter cubes

Word games are fun, right? There are lots of games in which players build words from letters on cubes. The gola of this problem is simple^ given a set of letter cubes and a group of words, write a program to determine if each word can or cannot be constructed from the cubes. Your program may re-order and omit cubes as necessary to form words. Also, each of the cubes can only have one side facing up, so only one letter from each of the cubes may be used for any particular word.

Input
The first line of input indicates the number of cubes, N, from three to nine. The next N lines give the letters that appear on the cubes. The following line indicates the number of words, M, from three to ten. And then the next M lines have one word each.

```
4
S X O C A D
O S N Q K M
B G D E L R
U E W I F P
10
CODE
WARS
LONG
DOG
SING
BRING
GLUE
IN
BAD
FORK
```

Output
For each word, the program must print the word and say whether it can or cannot be formed using the cubes.

```
CODE can be formed
WARS can be formed
LONG CANNOT be formed
DOG can be formed
SING can be formed
BRING CANNOT be formed
GLUE CANNOT be formed
IN can be formed
BAD CANNOT be formed
FORK can be formed
```

10. RoboRumble

RoboRumble is an exciting game where robots of different models fight each other in an arena. Each robot has unique specifications:
Name: The robot's model name (no spaces).
Category: Can be Mech, Drone, Cyborg, AI, Nano, or Quantum. A robot may have a vulnerability or armor against attacks from certain categories.
Vulnerability: If the robot is attacked by a robot of this category, the attack's damage is doubled.
Armor: If the robot is attacked by a robot of this category, the attack's damage is halved.
Integrity: A positive integer (up to 250) representing the robot's durability. When it drops to 0 or below, the robot is destroyed.
Abilities: Each robot has two abilities. Each ability has:
A name (no spaces)
An integer energy cost (1–5) to use the ability
An integer attack power (up to 200) indicating damage

At the start of the fight, robots have full integrity and zero energy. Robots take turns: each turn, the robot gains 1 energy and can use one ability if it has enough energy. If it can use both, it chooses the one with the higher attack power. The fight continues until one robot’s integrity drops to 0 or below. Write a program to simulate battles between various RoboRumble robots.

Input
The first line contains the number of robots.
Next lines contain robot details: Name Category Vulnerability Armor Integrity Ability1_Name EnergyCost Damage Ability2_Name EnergyCost Damage.
If a robot has no vulnerability or armor, it is indicated by X.
Then, a line with the number of battles to simulate.
Each battle is described by a line with the names of the two robots, in order of taking turns.

Example Input:

```
4
RoboX Mech Drone Nano 150 LaserBlast 2 50 RocketPunch 4 100
SkyDrone Drone Mech X 120 SonicBoom 1 30 AeroStrike 3 80
NanoCyborg Nano AI Quantum 100 HackPulse 1 40 NanoSlash 3 90
QuantumAI AI X Mech 80 MindBlast 2 60 QuantumBeam 4 120
3
RoboX SkyDrone
NanoCyborg RoboX
QuantumAI NanoCyborg
```

Output
For each battle, print:
<Winner> defeats <Loser> with <Remaining Integrity> integrity remaining.

```
RoboX defeats SkyDrone with 40 integrity remaining.
NanoCyborg defeats RoboX with 20 integrity remaining.
QuantumAI defeats NanoCyborg with 10 integrity remaining.
```