# Exercises on Strings, Streams & Linear Containers

# Problem 1 – Reverse words

Write a program that reads **lines** of space separated words from the standard input (until “end” word is met), reverse the order of all given words and outputs the result to the standard output (**words should be space-delimited**).

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Hello there  How are you end | you are How there Hello |
| end | “” (empty) |
| I did not quite  understand that  end | that understand quite not did I |

# Problem 2 – Numeral System

Write a program that reads the digits of a base-10 numeral system – exactly 10 unique non-space characters on a single line, representing the symbols used for the digits 0-9 (inclusively) – then reads two numbers represented in that numeral system, then prints their sum in that numeral system.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Explanation** |
| abcdefghij  cba ja | daa | a=0, b=1, …, j=9 cba = 210 ja = 90 90 + 210 = 300 = daa |
| 0123456789  42  512 | 554 | Numeral system matches normal base-10 (0=0, 1=1, …, 9=9) |
| qwertyuiop  popiu  tutiy | wtytyw | q=0, w=1, …, p=9 popiu = 98976 tutiy = 46475 98976 + 46475 = 145451 = wtytyw |

# Problem 3 – Minesweeper

You are given a N by M matrix (N and M are two integers entered on the console), in which the cells contain single characters – either a . (dot), or a ! (exclamation mark) – representing "empty" or "mined" positions.

Write a program that prints a N by M matrix, where each cell contains a number, representing how many adjacent cells, **including itself**, are "mined".

Each cell in a matrix has at most 8 adjacent cells – the cells directly above, below, to the left, to the right, as those diagonally – to the left and above, to the right and above, to the right and below and to the left and below.

### Examples

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 5 5  .....  ...!!  .....  .....  ..... | 00111  00111  00111  00000  00000 |  | 5 8  ........  ...!....  ....!...  ........  ...!.... | 00111000  00122100  00122100  00122100  00111000 |  | 3 3  !!!  !.!  !!! | 353  585  353 |

# Problem 4 – Rust

You are given a 10x10 matrix representing a metal square, which has begun to rust. There are 3 types of symbols in the matrix – a . (dot) means a healthy part of the metal, a # indicates a rust-resistant part, and a ! indicates a part that has begun to rust.

There may be **0, 1 or more** parts that has begun to rust.

The rust spreads from a rusty cell to healthy cells by "infecting" adjacent cells directly above, to the right, below and to the left of itself **(no diagonals)**, at the same time. The rust cannot infect cells that are indicated as rust-resistant. Let’s define the time it takes for all cells adjacent to a rusty cell to get infected as 1 unit.

After reading the matrix, read a single integer – the elapsed time in units (as defined above) – and print a matrix representing how the metal square will look after the rust has been acting on it for that amount of time

### Examples

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| ..........  ....!.....  ..........  ..........  ..........  ..........  ..........  ..........  ..........  ..........  4 | .!!!!!!!..  !!!!!!!!!.  .!!!!!!!..  ..!!!!!...  ...!!!....  ....!.....  ..........  ..........  ..........  .......... |  | ..........  ....!.....  ...###....  ..........  ..........  ..........  ..........  ..........  ..........  ..........  5 | !!!!!!!!!.  !!!!!!!!!!  !!!###!!!.  .!!!.!!!..  ..!...!...  ..........  ..........  ..........  ..........  .......... |  | !........!  ..........  ..........  ..........  ..........  ..........  ..........  ..........  ..........  !........!  5 | !!!!!!!!!!  !!!!!!!!!!  !!!!..!!!!  !!!....!!!  !!......!!  !!......!!  !!!....!!!  !!!!..!!!!  !!!!!!!!!!  !!!!!!!!!!  5 |

# Problem 5\* Ultimate Reverse words (it’s over 9000)

Write a program that reads **lines** of space separated words from the standard input (until “end” word is met), **reverse the order\*** of all given words (with the same size) and outputs the result to the standard output.

NOTE: the reverse of words follows a special procedure:

* Only words with the same character count are swapped;
* The first word containing 1 letter should be swapped with the last(Nth) word containing 1 letter. The second word with 1 letter should be swapped with the one before the last (Nth - 1) word container 1 letter.
* The same goes for 2 letter words, 3 letters words … N letter words;
* Punctuation should remain in the same place. (commas, dots, question marks, etc …);
* After the reverse all sentences should again start with capital letters. All other letters should be lowercase;

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Explanation** |
| Dude, what is this nightmare of a task! | Task, this of what nightmare is a dude! | First 4 letter word “Dude” swapped with last one “task”.  Second 4 letter word “what” swapped with one before last “this”  First 2 letter word “is” swapped with last one “of”  “Task” transformed to “task”;  “dude” transformed to “Dude” |
| Testing, testing, read all about it. | Testing, testing, read all about it. | Testing <-> testing |
| I can not believe - that a simple c++ program could have crashed so many times. | A c++ not crashed - many i simple can program times have believe so that could. | I <-> a  can <-> c++  many <-> that  times <-> could  crashed <-> bealive |
| Don’t hate the player – hate the game! | Don’t game the player – hate the hate! | hate <-> game  the <-> the |