Multiplication by hand

Submission deadline:	**2011-11-06 23:59:59**	533822.445 sec
Evaluation:	**0.0000**	
Max. assessment:	**3.0000** (Without bonus points)	
Submissions:	0 / 10 Free retries + 20 Penalized retries (-2 % penalty each retry)	
Advices:	0 / 2 Advices for free + 2 Advices with a penalty (-10 % penalty each advice)	

Your task is to develop a program that demonstrates multiplication by hand (as we learnt it in the basic school).

The input of the program are two non-negative integers to be multiplied.

The output of the program is the product of the two input numbers. Moreover, the output shall include the intermediate results, as shown in the examples below. The details of the formatting vary from school to school, therefore, the following guidelines shall be adhered:

- the output is indented by two spaces from the left,
- the width of the output is determined by the length of the result,
-
- the newline character always immediately follows the last printable character on a line, i.e. there is not any padding from the right,
- each nonzero digit in the second multiplier implies one intermediate result in the listing,
- zero digit in the second multiplier does not generate any extra line in the output. Instead, the zero is presented in the higher order intermediate result (one extra zero). This is demonstrated in the example `12345 x 10020`,
- the final result (last line) is followed by a newline.

The program must detect invalid input. If the input is not a number or if the input numbers are negative, the program must print an error message and terminate. The error message is followed by a newline character.

The output of your program must exactly match that of the reference. Again, use the enclosed archive and test your program with the provided input/expected output test data (see FAQ). Do not forget newlines, especially after the last line of the output.

Your program will be tested in a restricted environment. The testing environment limits running time and available memory. The exact time and memory limits are shown in the reference solution testing log. However, neither time nor memory limit could cause a problem in this simple program.

This problem is evaluated in a "bonus" mode. If your program passes all regular tests, it will be awarded nominal points. Standard `int` data type is sufficient for the regular tests. Both input numbers as well as the product fits into this data type in the regular tests. To pass the bonus test, your program must accept and provide correct answers for input numbers in the range of `long long int` data type.

Sample program output:

```
Enter two non-negative numbers:
1234 4321
Computation:
     1234
x    4321
---------
     1234
    2468
   3702
  4936
---------
  5332114
```

```
Enter two non-negative numbers:
12345 10020
Computation:
      12345
x     10020
-----------
     246900
  1234500
-----------
  123696900
```

```
Enter two non-negative numbers:
100 2000
Computation:
     100
x   2000
--------
  200000
--------
  200000
```

```
Enter two non-negative numbers:
1 50000
Computation:
      1
x 50000
-------
  50000
-------
  50000
```

```
Enter two non-negative numbers:
50000 1
Computation:
  50000
x     1
-------
  50000
-------
  50000
```

```
Enter two non-negative numbers:
222 0
Computation:
  222
x   0
-----
    0
-----
    0
```

```
Enter two non-negative numbers:
0 333
Computation:
    0
x 333
-----
    0
   0
```

```
  0
-----
    0
```

```
Enter two non-negative numbers:
65 -8
Invalid input.
```

```
Enter two non-negative numbers:
44 asdfg
Invalid input.
```

Sample run in the bonus test:

```
Enter two non-negative numbers:
270857638743723 785704153598122
Computation:
                  270857638743723
x                 785704153598122
---------------------------------
                  541715277487446
                 541715277487446
                270857638743723
              2166861109949784
             2437718748693507
            1354288193718615
            812572916231169
          1354288193718615
          270857638743723
        1083430554974892
     18960034712060610
    1354288193718615
   2166861109949784
  1896003471206061
---------------------------------
  212813971794722776382292088206
```

Sample data:		**Download**
Submit:	[redacted]	**Submit**

Reference