Specification for program “series”

Name

series – generate an additive series of numbers

Usage

series start end [stepsize]

Description

series prints the real numbers from **start** to **end**, one per line. **series** begins with **start** to which **stepsize** is repeatedly added or subtracted, as appropriate, to approach, possibly meet, but not pass **end**.
If all arguments are integers, only integers are produced in the output. The **stepsize** must be nonzero; if it is not specified, it is assumed to be of unit size (1). In all other cases, **series** prints an appropriate error message.

Example

To count from 1 to 100:

```
% series 1 100
```

To do the same, but backwards:

```
% series 100 1
```

Limitations

The reported number of significant digits is limited. If the ratio of the series range to the **stepsize** is too large, several numbers in a row will be equal.
The maximum length of a series is limited to the size of the maximum long integer that can be represented on the machine in use. Exceeding this value has undefined results.

Author

Gary Perlman

Worksheet for Training Exercise “series”

Subject identifier: ______

Nr.	V/I *	Equivalence class

*V = equivalence class for valid input, I = e. c. for invalid input

Table 1: Equivalence classes for program “series”

Nr.	Test case	Equiv. class	Expected output

Table 2: Tests based on equivalence classes for program “series”

Sample Solution for Training Exercise “series”

The following tables show examples of equivalence classes and test cases based on the equivalence classes for program “series”. These tables may be useful as example solutions.

Nr.	V/I *	Equivalence class
1	V	Argument count is 2
2	V	Argument count is 3
3	I	Argument count is neither 2 nor 3
4	I	First argument is not a number
5	I	Second argument is not a number
6	I	Third argument is not a number
7	V	All arguments are numbers
8	V	Start value is larger than end value
9	V	Start value is smaller than end value
10	V	Start value equals end value
11	I	Increment is zero
12	V	Increment is larger than abs (end – start)
13	V	Increment is smaller than abs (end – start)
14	V	All arguments are integer
15	V	At least one argument is a real

*V = equivalence class for valid input, I = e. c. for invalid input

Table 1: Equivalence classes for program “series”

Nr.	Test case	Equiv. class	Expected output
1	series 1	3	error message
2	series x 2	1, 4	error message
3	series 1 x	1, 5	error message
4	series 1 2 x	2, 6	error message
5	series 10 0	1, 7, 8, 14	10, 9, 8, ..., 1, 0
6	series 1 2 0.1	2, 7, 9, 13, 15	1.0, 1.1, 1.2, ..., 1.9, 2.0
7	series 7 7	1, 7, 10, 14	the number ‘7’
8	series 1 2 0	2, 7, 9, 11, 14	error message
9	series 2 1 0.1	2, 7, 8, 13, 15	2.0, 1.9, 1.8, ..., 1.1, 1.0
10	series 1 2 3	2, 7, 9, 12, 14	the number ‘1’

Table 2: Tests based on equivalence classes for program “series”