Multiplication tables

Preparation

To print strings in accordance with a very precise description of the layout, you can use the *str.format()* method in Python. This method can be called directly on a format string: formatstring.format(values). The formatstring is a string that contains so-called replacement fields, which are indicated by curly braces {}. The fields are replaced with arguments in the values in order. You can number the fields starting from 0 by putting the numbers between the curly braces. You can also reference fields by a name, which is put between the curly braces. In the values you can then use name=value (these pairs have to be the first arguments in the values). Inside the curly braces you can also put format specification by using : followed by certain characters to denote the format, in about the same way as you would use the % operator.

For example, the format specification {:s} indicates that a string must be placed in that position and the layout specification {:3d} indicates that an integer must be placed, lined out over 3 positions (completed by spaces on the left). The layout can be described with even more detail: the layout specification {:7.3f} for example, indicates that a floating point number must be written out with 3 digits after the comma, outlined on the right over 7 positions. For further expansions and details we refer to the text book (page 196, 4.4 formatted output for strings) or to the online Python manual. The example below illustrates the layout specifications that were described above.

```
>>> print('>>>{:s}<<<'.format('spam'))
>>>spam<<<
>>> print('>>>{:3d}<<<'.format(42))
>>> 42<<<
>>> print('>>>{:7.3f}<<<'.format(3.14))
>>>  3.140<<<
>>> print('{:s}\t{:d}\t{:5.3f}'.format('spam', 42, 3.14))
spam    42      3.140
```

Assignment

Write a program that prints the following multiplication grid:

```
    1   2   3   4   5   6   7   8   9  10  11  12
 1  1   2   3   4   5   6   7   8   9  10  11  12
 2  2   4   6   8  10  12  14  16  18  20  22  24
 3  3   6   9  12  15  18  21  24  27  30  33  36
 4  4   8  12  16  20  24  28  32  36  40  44  48
 5  5  10  15  20  25  30  35  40  45  50  55  60
 6  6  12  18  24  30  36  42  48  54  60  66  72
 7  7  14  21  28  35  42  49  56  63  70  77  84
 8  8  16  24  32  40  48  56  64  72  80  88  96
 9  9  18  27  36  45  54  63  72  81  90  99 108
10 10  20  30  40  50  60  70  80  90 100 110 120
11 11  22  33  44  55  66  77  88  99 110 121 132
12 12  24  36  48  60  72  84  96 108 120 132 144
```

Add enough spaces, so that all columns are exactly 3 digits wide and are separated by a column that is one space wide.

Voorbereiding

Om strings te kunnen uitschrijven volgens een zeer nauwkeurige beschrijving van de opmaak, kan je in Python gebruik maken van de *str.format()* methode. Deze methode kan rechtstreeks worden opgeroepen op een formaat string: formaatstring.format(waarden). De formaatstring is een string die zogenaamde vervangvelden bevat, die worden aangegeven met accolades {}. De velden worden in volgerde ingevuld met de argumenten in de waarden. Men kan de velden nummeren beginnende vanaf 0 door de nummers tussen de accolades te plaatsen. Naar velden kan ook verwezen worden via een naam die tussen de accolades geplaatst wordt, in de waarden zet men dan naam=waarde (deze paren moeten als eerste argumenten in de waarden voorkomen). Binnen de accolades kan men ook formaatspecificaties aangeven door gebruik te maken van een : gevolgd door bepaalde karakters die het formaat aangeven, ongeveer op dezelfde manier als wanneer de % operator gebruikt wordt.

De opmaakspecificatie {:s} geeft bijvoorbeeld aan dat op die positie een string moet geplaatst worden, en de opmaakspecificatie {:3d} geeft aan dat daar een integer moet komen, rechts uitgelijnd over 3 posities (links aangevuld met spaties). De opmaak kan zelfs nog veel gedetailleerder omschreven worden: de opmaakspecificatie {:7.3f} geeft bijvoorbeeld aan dat er een floating point getal moet uitgeschreven worden met 3 cijfers na de komma, rechts uitgelijnd over 7 posities. Voor verdere uitbreidingen en details verwijzen we naar het handboek (pagina 196, 4.4 formatted output for strings) of naar de online Python handleiding. Onderstaand voorbeeld illustreert de opmaakspecificaties die hierboven beschreven werden:

```
>>> print('>>>{:s}<<<'.format('spam'))
>>>spam<<<
>>> print('>>>{:3d}<<<'.format(42))
>>> 42<<<
>>> print('>>>{:7.3f}<<<'.format(3.14))
>>>  3.140<<<
>>> print('{:s}\t{:d}\t{:5.3f}'.format('spam', 42, 3.14))
spam    42      3.140
```

Opgave

Schrijf een programma dat het volgende vermenigvuldigingsrooster uitschrijft:

```
    1   2   3   4   5   6   7   8   9  10  11  12
 1  1   2   3   4   5   6   7   8   9  10  11  12
 2  2   4   6   8  10  12  14  16  18  20  22  24
 3  3   6   9  12  15  18  21  24  27  30  33  36
 4  4   8  12  16  20  24  28  32  36  40  44  48
 5  5  10  15  20  25  30  35  40  45  50  55  60
 6  6  12  18  24  30  36  42  48  54  60  66  72
 7  7  14  21  28  35  42  49  56  63  70  77  84
 8  8  16  24  32  40  48  56  64  72  80  88  96
 9  9  18  27  36  45  54  63  72  81  90  99 108
10 10  20  30  40  50  60  70  80  90 100 110 120
11 11  22  33  44  55  66  77  88  99 110 121 132
12 12  24  36  48  60  72  84  96 108 120 132 144
```

Voeg voldoende spatiëring in zodat alle kolommen juist 3 karakters breed zijn en van elkaar gescheiden worden door een kolom van één spatie breed.