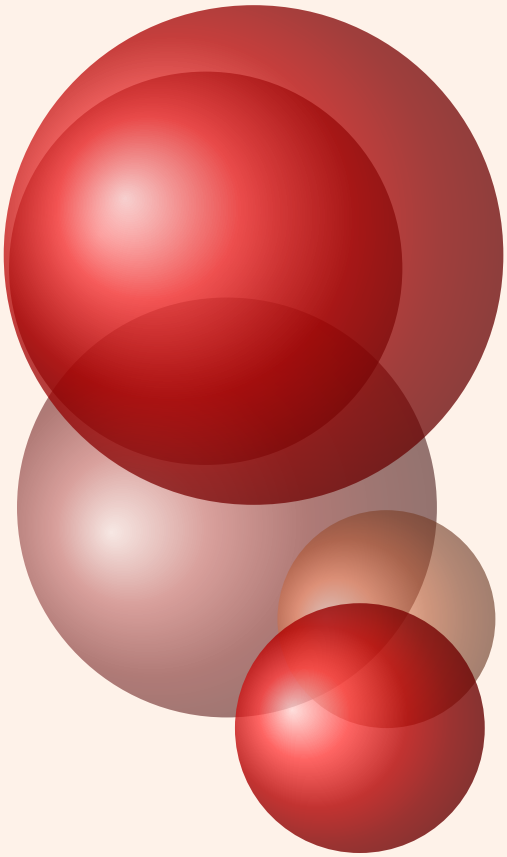
tkz-berge.sty v 1.00 c

AlterMundus



Alain Matthes

May 26, 2011

http://altermundus.fr http://altermundus.com

tkz-berge.sty

AlterMundus

Alain Matthes

The package tkz-berge.sty is a collection of some useful macros if you want to draw some classic graphs of the graph theory or to make others graphs. The kind of graphs that I will present, are sometimes called combinatorial graphs to distinguish them from the graphs of functions. Often, the word graph is short for graph of a function. A combinatorial graph is a very simple structure, a bunch of dots, some of which are connected by lines. Some of graphs have names, sometimes inspired by the graph's topology, and sometimes after their discoverer.

Why tkz-berge.sty ?

Claude Berge (1926 – 2002) was a French mathematician, recognized as one of the modern founders of combinatorics and graph theory. He played a major role in the renaissance of combinatorics and he is remembered for his famous conjecture on perfect graphs, solved some months after his death.

☞Firstly, I would like to thank **Till Tantau** for the beautiful LATEX package, namely TikZ.
☞I am grateful to **Michel Bovani** for providing the **fourier** font.
☞I received much valuable advice and guidance on Graph Theory from **Rafael Villarroel** http://graphtheoryinlatex.blogspot.com/.
☞The names of graphs can be found here MathWorld - SimpleGraphs by E.Weisstein

Contents

List of the main macros :

- **\grEmptyCycle**
- **\grEmptyPath**
- **\grEmptyStar**
- **\grEmptyGrid**
- **\grEmptyLadder**
- **\EdgeInGraphFromOneToComp**
- **\EdgeInGraphLoop**
- **\EdgeInGraphSeq**
- **\EdgeInGraphMod**
- **\EdgeInGraphMod***
- **\grCompleteBipartite**
- **\EdgeInGraphModLoop**
- **\EdgeIdentity**
- **\EdgeIdentity***
- **\EdgeFromOneToAll**
- **\EdgeFromOneToSeq**
- **\EdgeFromOneToSel**
- **\EdgeFromOneToComp**
- **\EdgeMod**
- **\EdgeMod***
- **\EdgeDoubleMod**
- **\grPath**
- **\grCycle**
- **\grComplete**
- **\grCirculant**
- **\grStar**
- **\grSQCycle**
- **\grWheel**
- **\grLadder**
- **\grPrism**
- **\grCompleteBipartite**
- **\grTriangularGrid**
- **\grLCF**
- **\grWriteExplicitLabels**
- **\grWriteExplicitLabel**
- **\AssignVertexLabel**

See the document "NamedGraph" for all the classic named graphs that you can draw with the package tkz-berge.sty.

SECTION 1

Installation

1.1 How to install the package berge.sty

It is possible that when you read this document, **tkz-berge** is present on the **CTAN**[1] servers. If **tkz-berge** is not still a part of your distribution, this chapter shows you how to install it.

1.2 With TeXLive under OS X and Linux

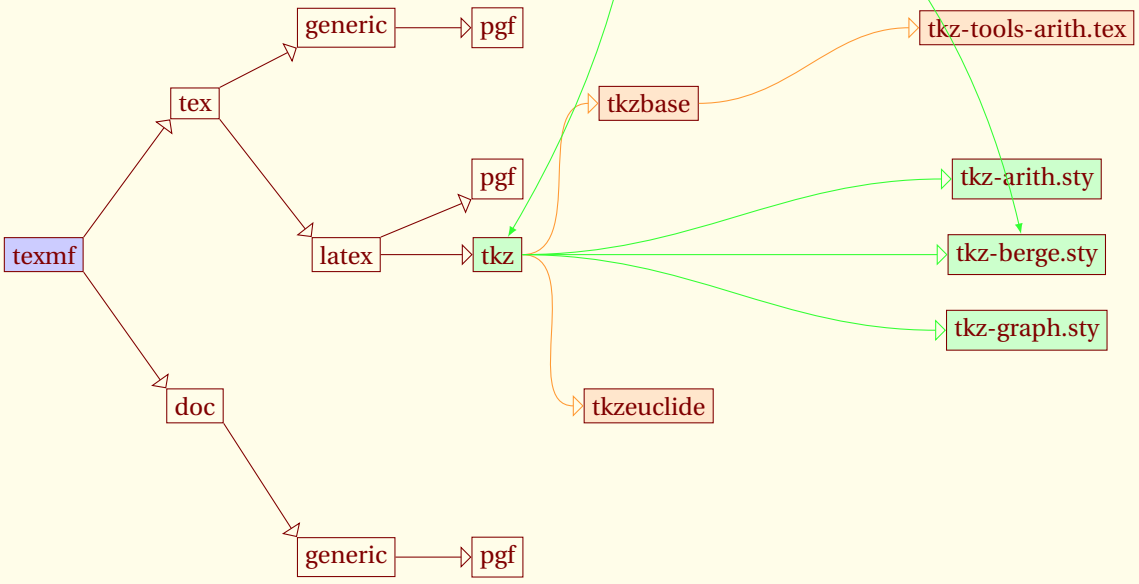
You could simply create a folder (directory) tkz which path is : texmf/tex/latex/tkz .
texmf is generally the personnal folder. For example the paths of this folder on my two computers are

- with OS X **/Users/ego/Library/texmf** ;
- with Ubuntu **/home/ego/texmf** .

If you choose a custom location for your files, I suppose that you know why! The installation that I propose, is valid only for one user.

1. Store the files tkz-arith.sty, tkz-graph.sty et tkz-berge.sty in the folder prof .
2. Open a terminal, then type `sudo texhash`
3. Check that **xkeyval** version 2.5 or more, and **TikZ 2.1** are installed because they are obligatory.

My folder texmf is structured as in the diagram below because I use the **CVS**[2] version of **TikZ**. You don't need all the **pgf** folders.



1 **tkz-berge** is not still a part of **TeXLive** but it will be soon possible to install it with **tlmgr**
2 You can find the cvs version here : http://www.texample.net/tikz/builds/ without CVS or here with CVS http://sourceforge.net/projects/pgf/

1.3 How to work with the tkz-LaTeX-package under Windows?

Download and install the following files (if not yet done):

1. the LaTeX-system MiKTeX from

   http://www.miktex.org/.

   What file you need (e.g. `basic-miktex-2.7.2904.exe`) and how to install this program is explained there in the "Download" section of the respective version (current version is 2.7). In general and as usual in windows, you run the setup process by starting the setup file :
   (e.g.`basic-miktex-2.7.2904.exe`).

2. Till Tantau's LaTeX-package `pgf-tikZ` from

   http://sourceforge.net/projects/pgf/

   "For MiKTeX, use the update wizard [of MiKTeX] to install the (latest versions of the) packages called `pgf`, `xcolor`, and `xkeyval`." (cited from the pgf manual, contained in the files downloaded).

3. the sty-files and the doc-files of Alain's tkz-package from the CTAN servers or

   http://www.altermundus.fr/pages/download.html.

   or

   http://altermundus.com/pages/downloads/index.html.

   To add the files to MiKTeX:

   - add a directory `prof` in the directory `[MiKTeX-dir]/tex/latex`, e.g. in windows explorer,
   - copy the sty-files in this directory `prof`,
   - update the MiKTeX system, ether by running in a DOS shell the command
     "`mktexlsr -u`"
     or by clicking
     "`Start/Programs/Miktex/Settings/General`", then push the button `Refresh FNDB`.

SECTION 2

Macros and Vertices

2.1 \grEmptyCycle

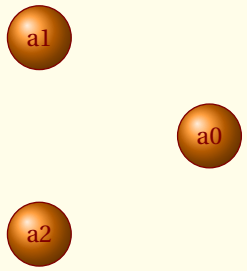
\grEmptyCycle[⟨*local options*⟩]{⟨*order*⟩}

Arguments	Definition
order	order of the graph

Options	default	definition
RA	4	radius circle
prefix	a	prefix for vertices
Math	false	math mode

The number of nodes in a graph is called its order. The argument "order" is an integer superior to 1. RA *defines the radius of the circle.*

2.1.1 Empty Cycle

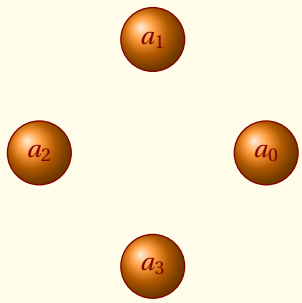



```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \grEmptyCycle[RA=1.5]{3}
\end{tikzpicture}
```

2.1.2 Empty Cycle and \SetVertexNoLabel

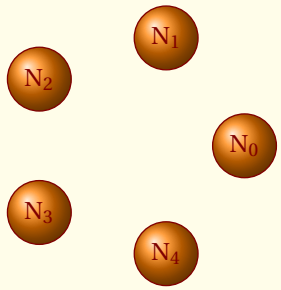
```
\begin{tikzpicture}
  \SetVertexNoLabel
  \GraphInit[vstyle=Shade]
  \grEmptyCycle[RA=1.5]{2}
\end{tikzpicture}
```

2.1.3 Empty Cycle and Math



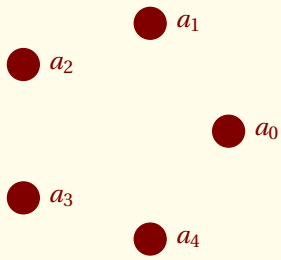

```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \grEmptyCycle[Math,RA=1.5]{4}
\end{tikzpicture}
```

2.1.4 Empty Cycle, \SetVertexMath and prefix



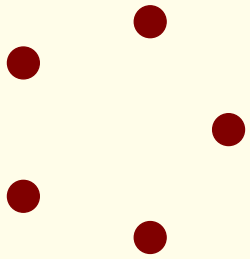

```
\begin{tikzpicture}
  \SetVertexMath
  \GraphInit[vstyle=Shade]
  \grEmptyCycle[prefix=N,RA=1.5]{5}
\end{tikzpicture}
```

2.1.5 Empty Cycle and Classic style




```
\begin{tikzpicture}
  \SetVertexMath
  \GraphInit[vstyle=Classic]
  \grEmptyCycle[RA=1.5]{5}
\end{tikzpicture}
```

2.1.6 Empty Cycle and Simple style



```
\begin{tikzpicture}
  \GraphInit[vstyle=Simple]
  \grEmptyCycle[RA=1.5]{5}
\end{tikzpicture}
```

2.2 \grEmptyPath

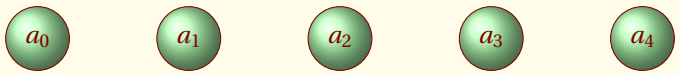
\grEmptyPath[⟨*local options*⟩]{⟨*order*⟩}

Arguments	Definition
order	order of the graph

options	default	definition
RA	4 cm	distance between two vertices
RS	? cm	distance between the first line and the new one
prefix	a	prefix for vertices
Math	false	math mode

Order *is the number of nodes.* RA *defines the radius of the circle.* RS *defines the distance between the graph and the baseline.*

2.2.1 Empty Path, RA and Math

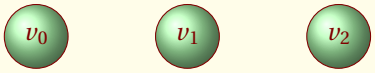



```
\begin{tikzpicture}
   \grEmptyPath[Math,RA=2]{5}
\end{tikzpicture}
```

2.2.2 Empty Path, RA and prefix

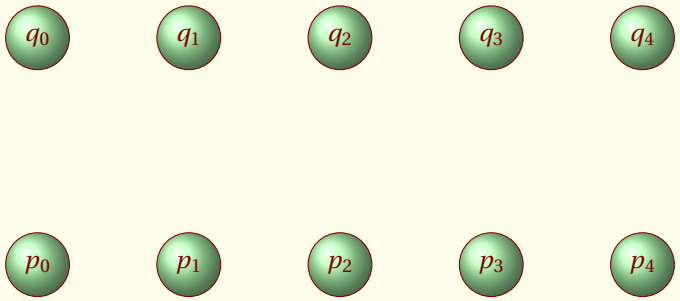

```
\begin{tikzpicture}
   \grEmptyPath[prefix=h,RA=2]{6}
\end{tikzpicture}
```

2.2.3 Empty Path, vertical path with form=2

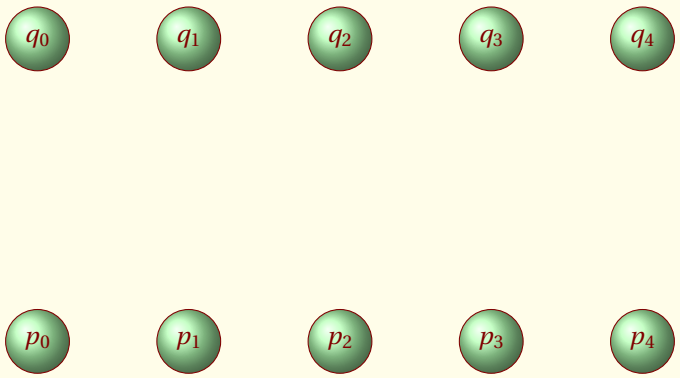



```
\begin{tikzpicture}
 \grEmptyPath[form=2,prefix=v,RA=2]{3}
\end{tikzpicture}
```

2.2.4 Two Empty Paths

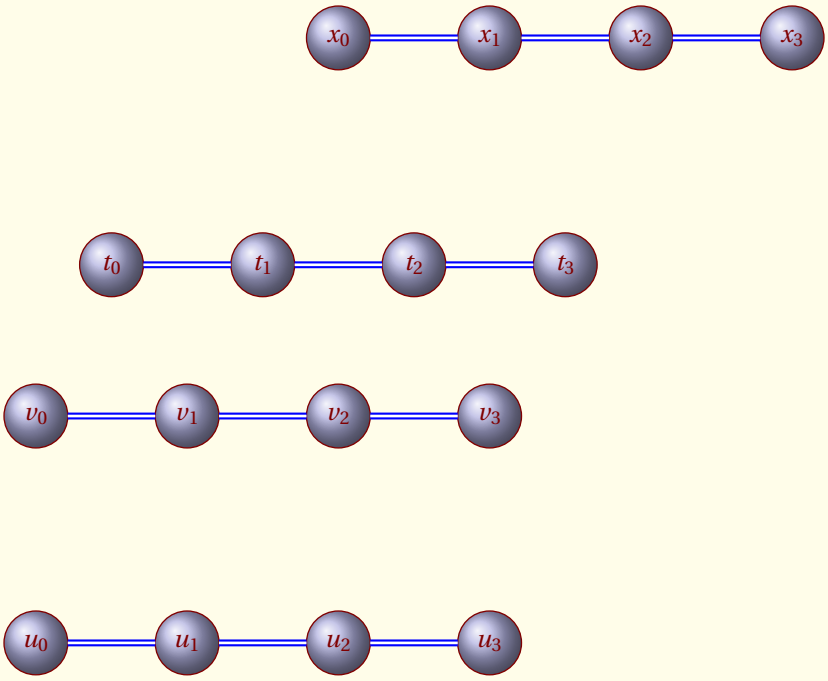



```
\begin{tikzpicture}
    \grEmptyPath[Math,prefix=p,RA=2,RS=0]{5}
    \grEmptyPath[Math,prefix=q,RA=2,RS=3]{5}
\end{tikzpicture}
```




```
\begin{tikzpicture}
    \grEmptyPath[Math,prefix=p,RA=2,RS=0,form=2]{5}
    \grEmptyPath[Math,prefix=q,RA=2,RS=4,form=2]{5}
\end{tikzpicture}
```

2.2.5 How to move a graph ?




```
\begin{tikzpicture}
     \grPath[Math,prefix=u,RA=2,RS=0]{4}
     \grPath[Math,prefix=v,RA=2,RS=3]{4}
     \begin{scope}[xshift=1 cm]
      \grPath[Math,prefix=t,RA=2,RS=5]{4}
     \end{scope}
     \begin{scope}[shift={(4 cm,8cm)}]
      \grPath[Math,prefix=x,RA=2,RS=0]{4}
     \end{scope}
\end{tikzpicture}
```

2.3 Empty Star

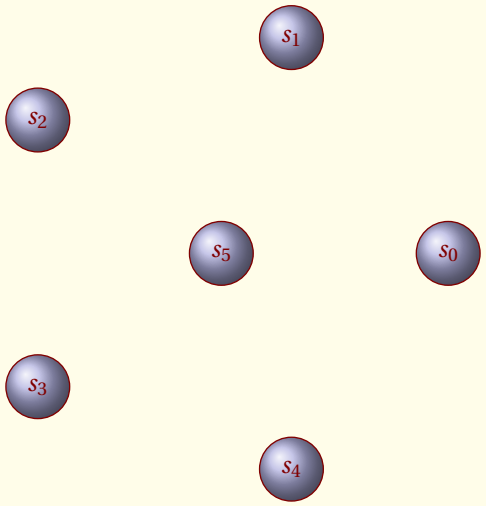
\grEmptyStar[⟨*local options*⟩]{⟨*order*⟩}

Arguments	Definition
order	order of the graph

options	default	definition
RA	4 cm	radius circle
prefix	a	prefix for vertices
Math	false	math mode

RA *defines the radius of the circle.* order *is an integer and it's the order of the graph.*

2.3.1 Empty Star




```
\begin{tikzpicture}
  \SetVertexMath
  \grEmptyStar[prefix=s,RA=3]{6}
\end{tikzpicture}
```

2.4 Empty Grid

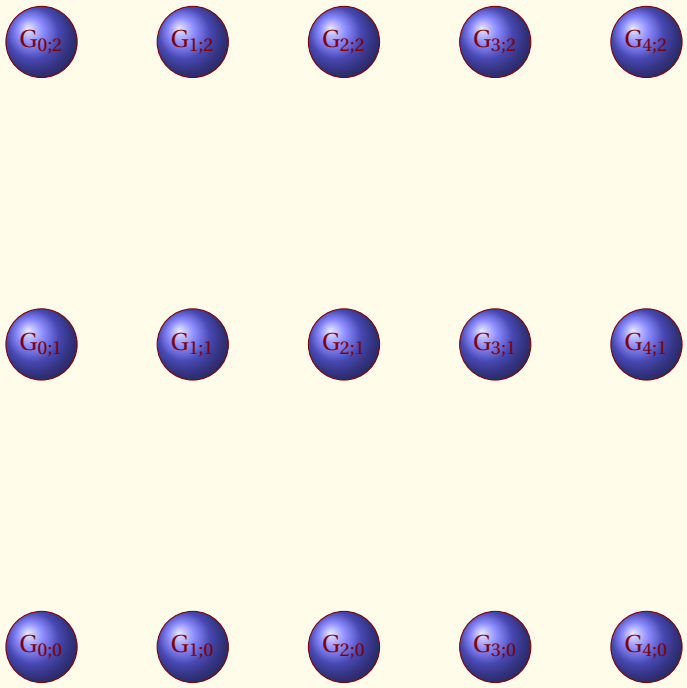
\grEmptyGrid[⟨*local options*⟩]{⟨*c*⟩}{⟨*r*⟩}

Arguments	Definition
r	number of rows
c	number of columns

options	default	definition
RA	4 cm	distance between two columns
RB	3 cm	distance between two rows
prefix	3 cm	distance between two rows
Math	false	math mode

c *and* r *are integers.*

2.4.1 Prefix




```
\begin{tikzpicture}
   \tikzset{VertexStyle/.style ={shape        = circle,
                                 shading      = ball,
                                 ball color   = Blue!60,%
                                 minimum size = 24pt,%
                                 draw}}
  \SetVertexMath
  \grEmptyGrid[prefix=G,RA=2,RB=4]{5}{3}
\end{tikzpicture}
```

2.5 Empty Ladder

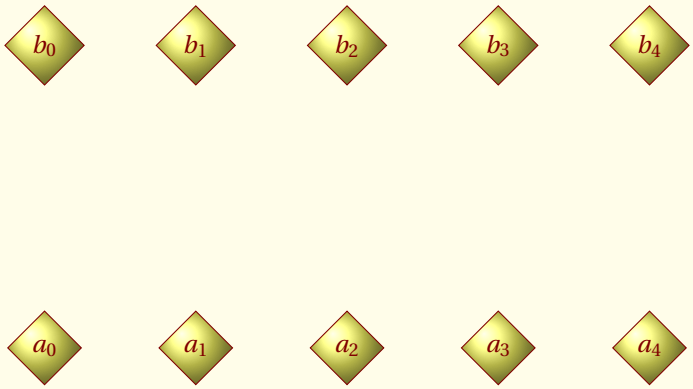
\grEmptyLadder[⟨*local options*⟩]{⟨c⟩}

Arguments	Definition
c	number of columns.

options	default	definition
RA	4 cm	distance between two columns
RB	3 cm	distance between two rows
prefix	a	prefix for vertices
prefix	b	prefix for vertices
Math	false	math mode

c *is an integer. There are only two rows with different prefix.*

2.5.1 Empty Ladder




```
\begin{tikzpicture}
   \tikzset{VertexStyle/.style ={shape        = diamond,
                                 shading      = ball,
                                 ball color   = yellow!60,%
                                 minimum size = 24pt,%
                                 draw}}
   \SetVertexMath
   \grEmptyLadder[RA=2,RB=4]{5}
\end{tikzpicture}
```

SECTION 3

Macros and Edges in a graph

3.1 Edge in a graph from one vertex \EdgeInGraphFromOneToComp

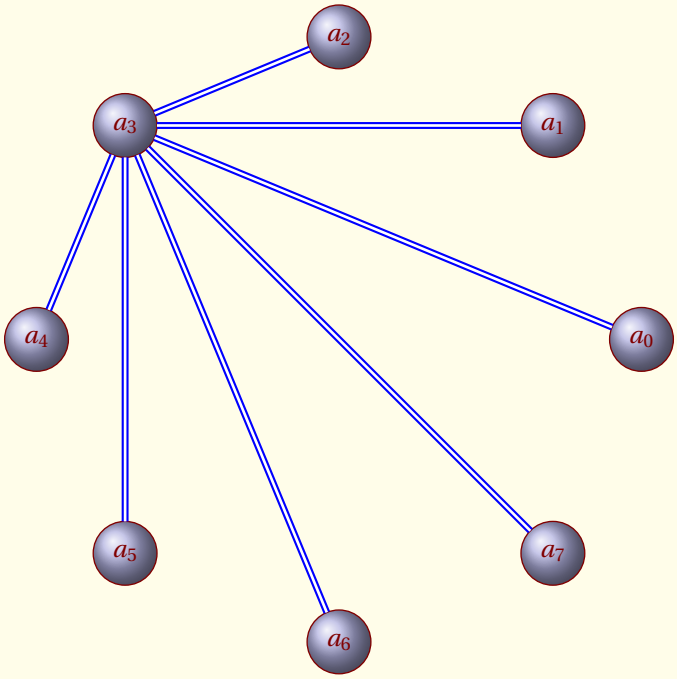
\EdgeInGraphFromOneToComp[⟨*local options*⟩]{⟨*prefix*⟩}{⟨*order*⟩}{⟨*from*⟩}

Arguments	Definition
order	order of the graph

options	default	definition
RA	4	radius circle
prefix	a	prefix for vertices
Math	false	math mode

This macro works on an unique graph. `from` *is integer.* `EdgeInGraph` *designs a macro that works only in a graph defined by a prefix. The result is some edges between the vertex* `from` *and the others vertices.*

3.1.1 Empty Cycle



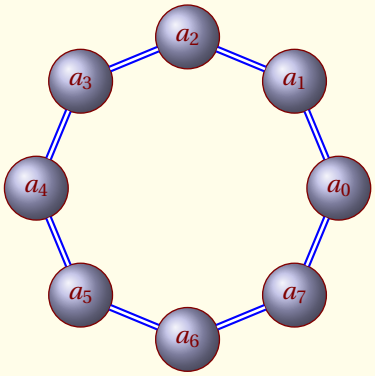

```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \grEmptyCycle[RA=4,prefix=a]{8}%
  \EdgeInGraphFromOneToComp{a}{8}{3}
\end{tikzpicture}
```

3.2 Edges in a graph - a loop \EdgeInGraphLoop

\EdgeInGraphLoop{⟨*prefix*⟩}{⟨*order*⟩}

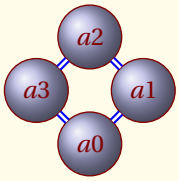
This macro is useful with vertices on a circle . `order` *in an integer.*

3.2.1 Empty Cycle




```
\begin{tikzpicture}
\GraphInit[vstyle=Shade]
\grEmptyCycle[RA=2,prefix=a]{8}%
\EdgeInGraphLoop{a}{8}
\end{tikzpicture}
```

3.2.2 Empty Cycle



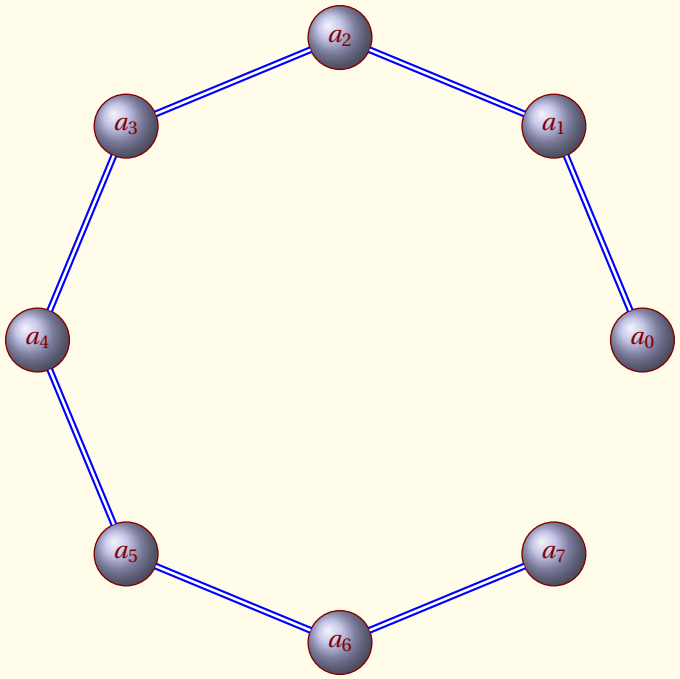

```
\begin{tikzpicture}[node distance=4cm]
 \GraphInit[vstyle=Shade]
 \Vertices{square}{a0,a1,a2,a3}
 \EdgeInGraphLoop{a}{4}
\end{tikzpicture}
```

3.3 Edges in a graph - a loop \EdgeInGraphLoop*

\EdgeInGraphLoop*{⟨*prefix*⟩}{⟨*order*⟩}

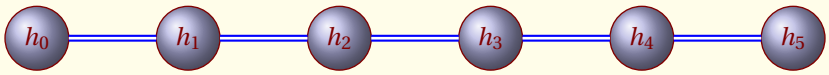
Not exactly a loop, there is no edge between the first and the last vertex.

3.3.1 Empty Cycle




```
\begin{tikzpicture}
  \GraphInit[vstyle=Art]
  \grEmptyCycle[RA=4,prefix=a]{8}%
  \EdgeInGraphLoop*{a}{8}
\end{tikzpicture}
```

3.3.2 Empty Path



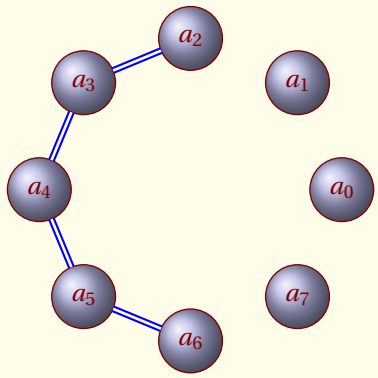

```
\begin{tikzpicture}
     \grEmptyPath[prefix=h,RA=2,RS=2]{6}
     \EdgeInGraphLoop*{h}{6}
\end{tikzpicture}
```

3.4 Sequence of edges in a graph \EdgeInGraphSeq

\EdgeInGraphSeq{⟨*prefix*⟩}{⟨*start*⟩}{⟨*end*⟩}

This macro gives a sequence of edges between start *and* end.
start *and* end *are two integers.*

3.4.1 EdgeInGraphSeq



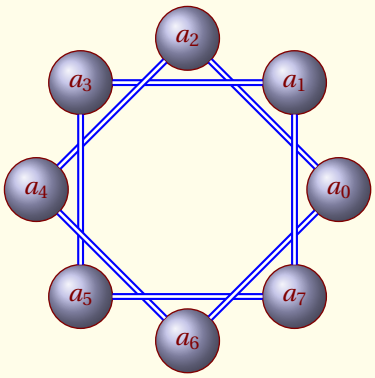

```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \grEmptyCycle[RA=2,prefix=a]{8}%
  \EdgeInGraphSeq{a}{2}{5}
\end{tikzpicture}
```

3.5 Edges in a graph \EdgeInGraphMod

\EdgeInGraphMod{⟨*prefix*⟩}{⟨*order*⟩}{⟨*add*⟩}

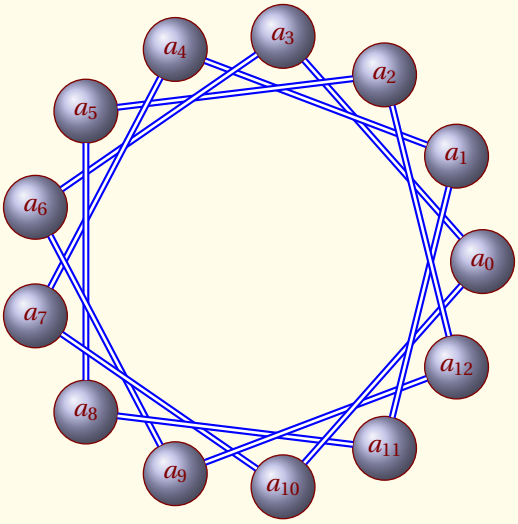
This macro works on an unique graph. Edges between v_i and v_j with i in $0,...,(\#2-1)$ and $j = Mod(i+\#3,\#2)$.
#2 = `order` *and #3 =* `add`.
`Mod` *is like* `mod` *but the result is a positive integer.*

3.5.1 EdgeInGraphMod




```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \grEmptyCycle[RA=2,prefix=a]{8}%
  \EdgeInGraphMod{a}{8}{2}
\end{tikzpicture}
```

3.5.2 EdgeInGraphMod 2



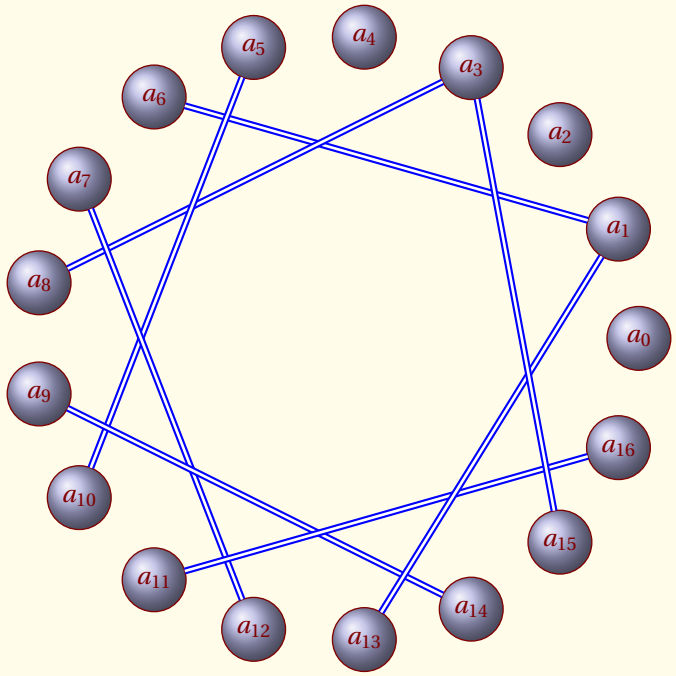

```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \grEmptyCycle[RA=3,prefix=a]{13}%
  \EdgeInGraphMod{a}{13}{3}
\end{tikzpicture}
```

3.6 Edges in a graph \EdgeInGraphMod*

\EdgeInGraphMod*{⟨*prefix*⟩}{⟨*order*⟩}{⟨*add*⟩}{⟨*start*⟩}{⟨*step*⟩}

Edges between v_i and v_j with i in #4, #4 + #5, ..., (#2 − 1) *and* j = *Mod(i+#3,#2)*
#2 = order, #3 = add, #4 = start, #5 = step.

3.6.1 EdgeInGraphMod*



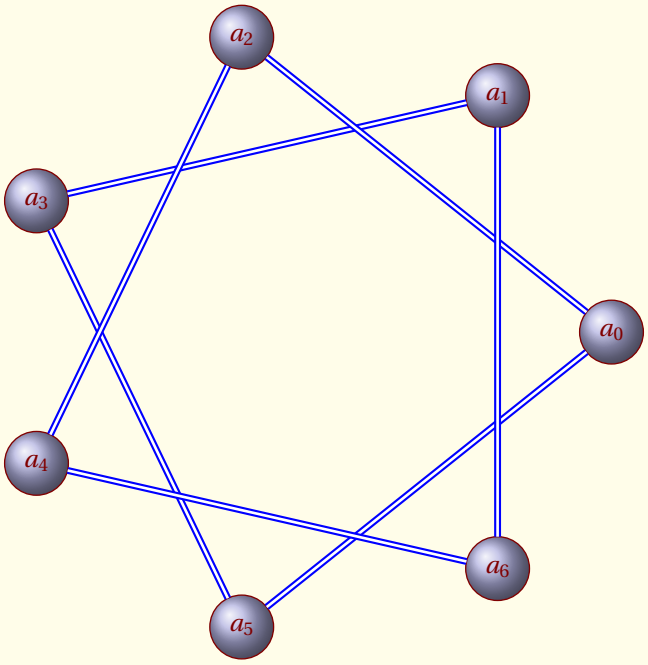

```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \grEmptyCycle[prefix=a]{17}%
  \EdgeInGraphMod*{a}{17}{5}{1}{2}
\end{tikzpicture}
```

3.7 Edges in a graph \EdgeInGraphModLoop

\EdgeInGraphModLoop{⟨prefix⟩}{⟨order⟩}{⟨add⟩}{⟨start⟩}

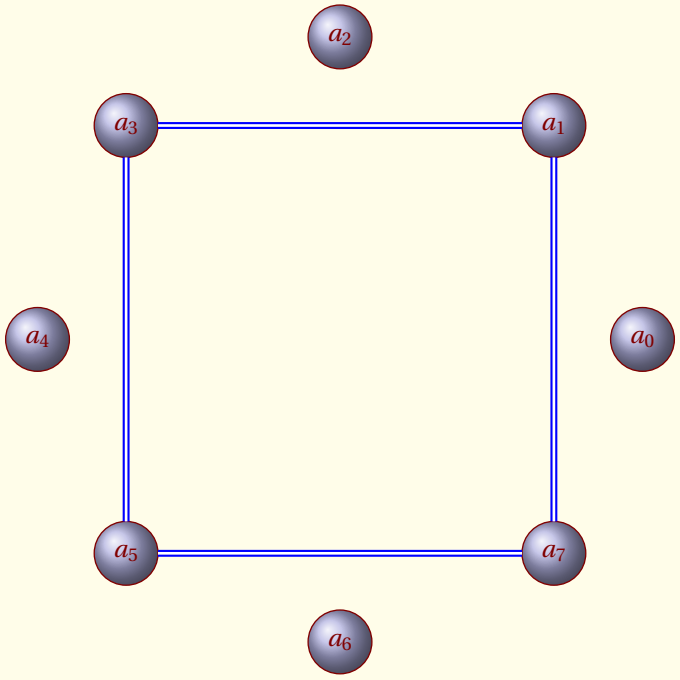
order, add *and* start *are integers.*
Edges between v_i *and* v_j *with i from* #4, j = *Mod(i+#3,#2) and then* $i = j$ *until* j = #4
#2 = order, *#3* = add *and #4* = start.

3.7.1 EdgeInGraphModLoop




```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \grEmptyCycle[RA=4]{7}
  \EdgeInGraphModLoop{a}{7}{2}{1}
\end{tikzpicture}
```

3.7.2 EdgeInGraphModLoop



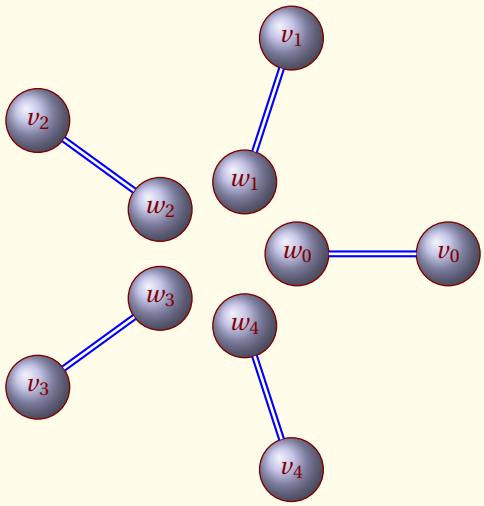

```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \grEmptyCycle[RA=4]{8}
  \EdgeInGraphModLoop{a}{8}{2}{1}
\end{tikzpicture}
```

3.8 Edges between two graphs with the same order \EdgeIdentity

\EdgeIdentity{⟨*prefix1*⟩}{⟨*prefix2*⟩}{⟨*order*⟩}

`order` *is an integer. This macro gives edges between two graphs.*
Edges between v_i *and* v_j *with* $i = j$ *in* $0,...,(\#3-1)$.
#3 = `order`.

3.8.1 EdgeIdentity



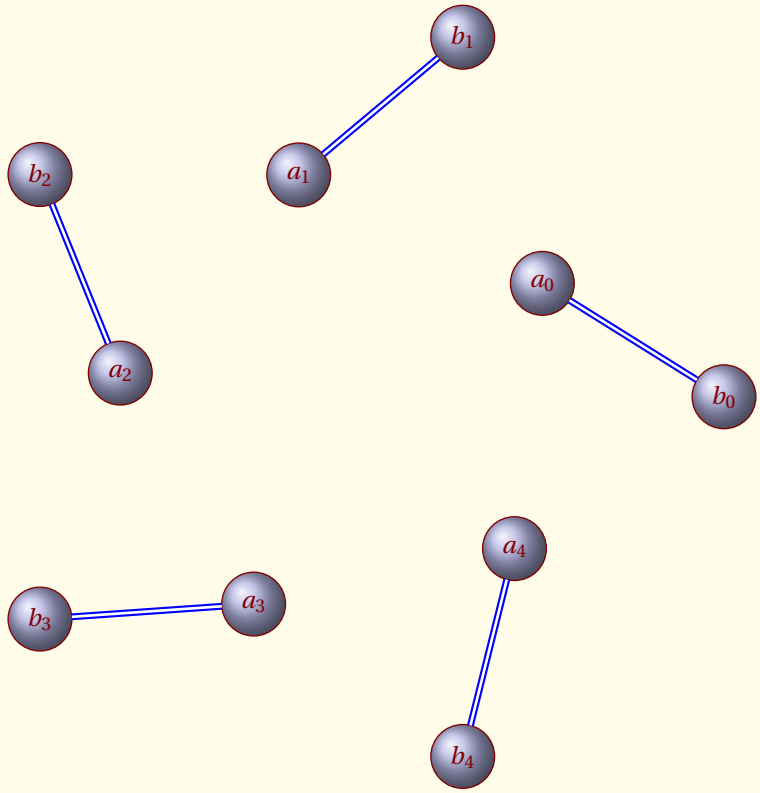

```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
    \grEmptyCycle[prefix=v,RA=3]{5}
    \grEmptyCycle[prefix=w,RA=1]{5}
    \EdgeIdentity{v}{w}{5}
\end{tikzpicture}
```

3.9 Edges between two graphs with the same order \EdgeIdentity*

\EdgeIdentity*{⟨*prefix1*⟩}{⟨*prefix2*⟩}{⟨*list*⟩}

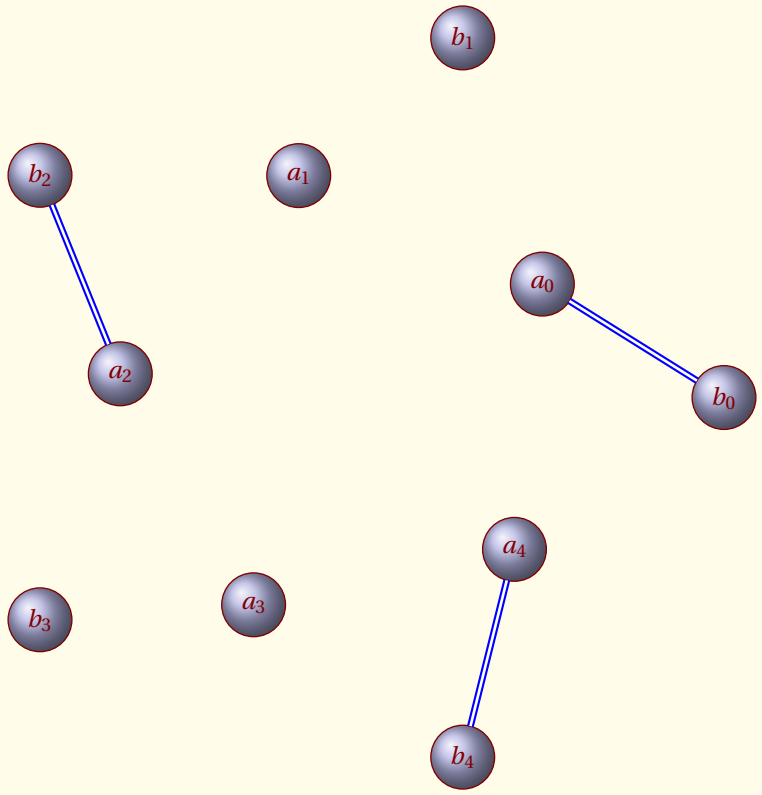
`list` *is a list of integers. This macro gives edges between two graphs.*
Edges between v_i *and* v_j *with* $i = j$ *in* `list`.

3.9.1 EdgeIdentity*




```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \begin{scope}[rotate=30]
  \grEmptyCycle[RA=3,prefix=a]{5}%
  \end{scope}
  \grEmptyCycle[RA=5,prefix=b]{5}%
   \EdgeIdentity*{a}{b}{0,...,4}
\end{tikzpicture}
```

3.9.2 EdgeIdentity*



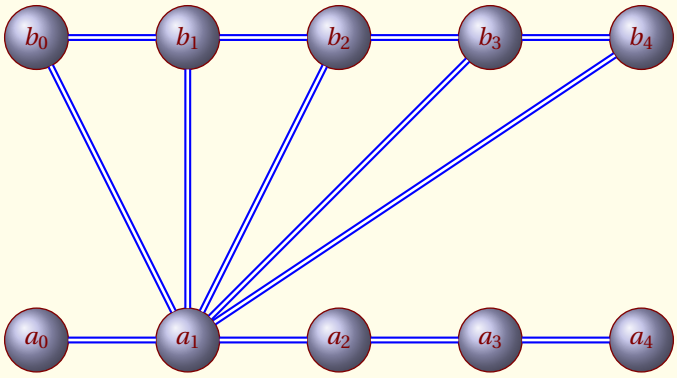

```
\begin{tikzpicture}
 \GraphInit[vstyle=Shade]
  \begin{scope}[rotate=30]
  \grEmptyCycle[RA=3,prefix=a]{5}%
  \end{scope}
  \grEmptyCycle[RA=5,prefix=b]{5}%
   \EdgeIdentity*{a}{b}{0,2,4}
\end{tikzpicture}
```

3.10 Edges between two graphs \EdgeFromOneToAll

\EdgeFromOneToAll{⟨prefix1⟩}{⟨prefix2⟩}{⟨from⟩}{⟨order⟩}

The graphs must to have the same order. `from` *and* `order` *are integers.*

3.10.1 EdgeFromOneToAll



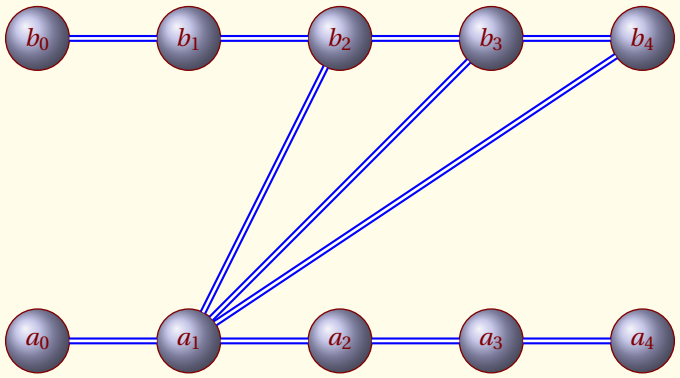

```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \grPath[form=1,RA=2,RS=0]{5}
  \grPath[form=1,prefix=b,RA=2,RS=4]{5}
  \EdgeFromOneToAll{a}{b}{1}{5}
\end{tikzpicture}
```

3.11 Edges between two graphs \EdgeFromOneToSeq

\EdgeFromOneToSeq{⟨prefix1⟩}{⟨prefix2⟩}{⟨from⟩}{⟨start⟩}{⟨end⟩}

from, start *and* end *are integers. This macro builds edges between the vertex with an indice* from *through the vertices with an indice in the sequence* start,...,end.

3.11.1 EdgeFromOneToSeq



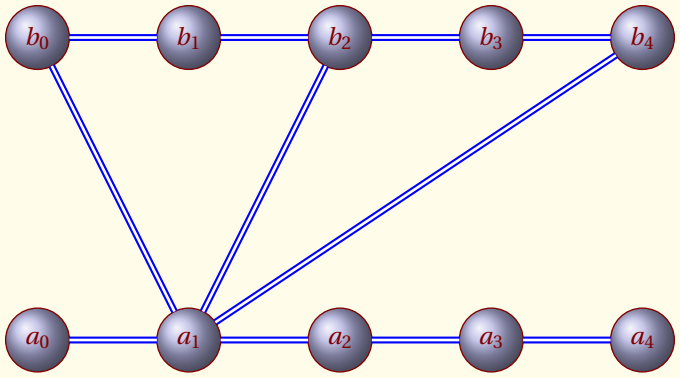

```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \grPath[form=1,RA=2,RS=0]{5}
  \grPath[form=1,prefix=b,RA=2,RS=4]{5}
  \EdgeFromOneToSeq{a}{b}{1}{2}{4}
\end{tikzpicture}
```

3.12 Edges between two graphs \EdgeFromOneToSel

\EdgeFromOneToSel{⟨*prefix1*⟩}{⟨*prefix2*⟩}{⟨*from*⟩}{⟨*list*⟩}

This macro builds edges between the vertex with an indice `from` *through the vertices with an indice in the list* `list`.

3.12.1 EdgeFromOneToSel



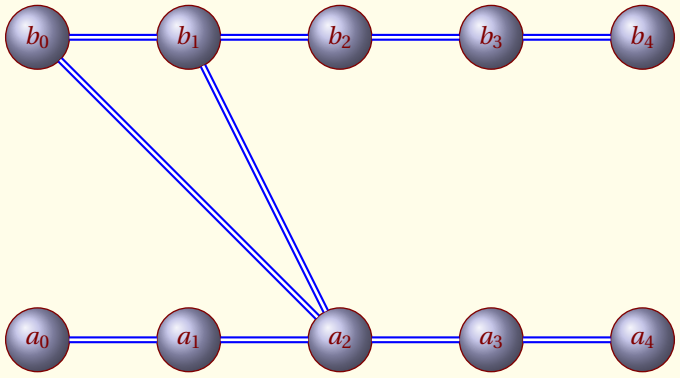

```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \grPath[form=1,RA=2]{5}
  \grPath[form=1,prefix=b,RA=2,RS=4]{5}
  \EdgeFromOneToSel{a}{b}{1}{0,2,4}
\end{tikzpicture}
```

3.13 Edges between two graphs \EdgeFromOneToComp

\EdgeFromOneToComp{⟨*prefix1*⟩}{⟨*prefix2*⟩}{⟨*from*⟩}{⟨*order2*⟩}

This macro builds edges between the vertex with an indice `from` *through all the vertices of the second graph, except the vertex with an indice* `from`.

3.13.1 EdgeFromOneToComp



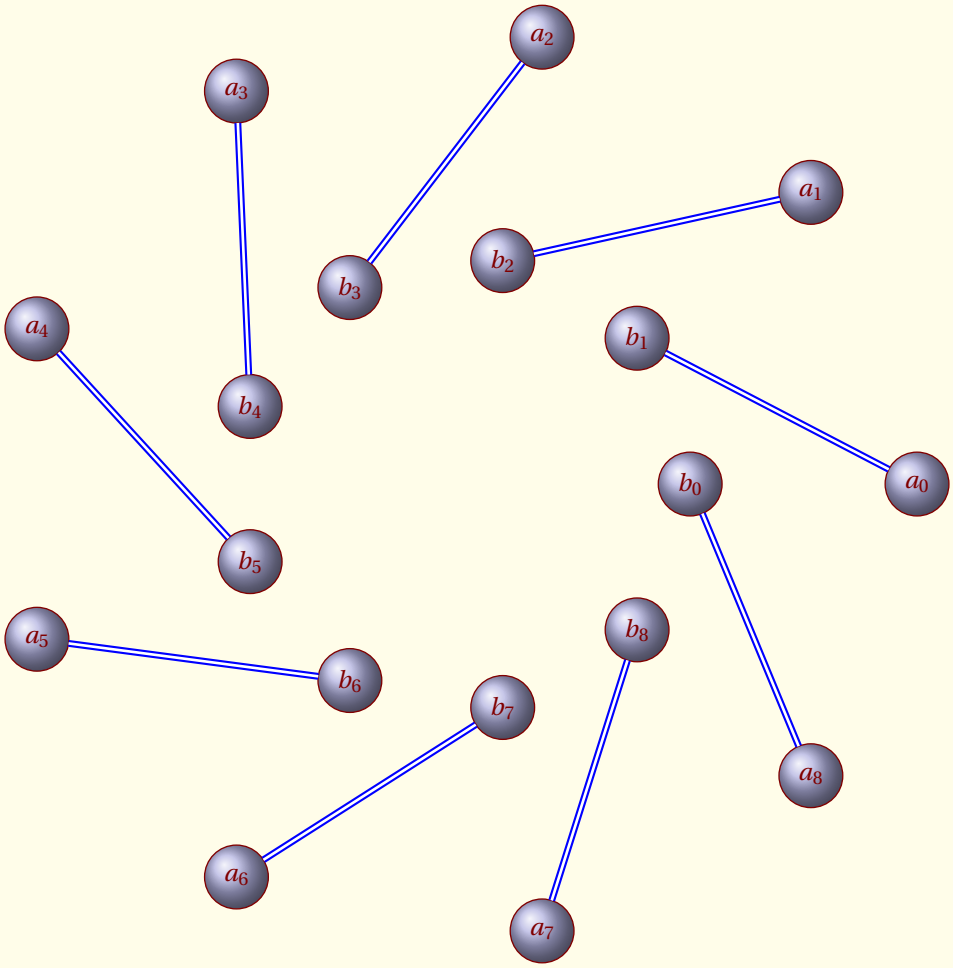

```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \grPath[form=1,RA=2,RS=0]{5}
  \grPath[form=1,prefix=b,RA=2,RS=4]{5}
  \EdgeFromOneToComp{a}{b}{2}{3}
\end{tikzpicture}
```

3.14 Edges between two graphs \EdgeMod

\EdgeMod{⟨*prefix1*⟩}{⟨*prefix2*⟩}{⟨*order*⟩}{⟨*step*⟩}

This macro works on two graphs with the same order. We get edges between v_i and v_j with i in $0,...,(\#2-1)$ and j = Mod(i+#4,#3).
#3 = `order` *and #4* = `step`.

3.14.1 EdgeMod



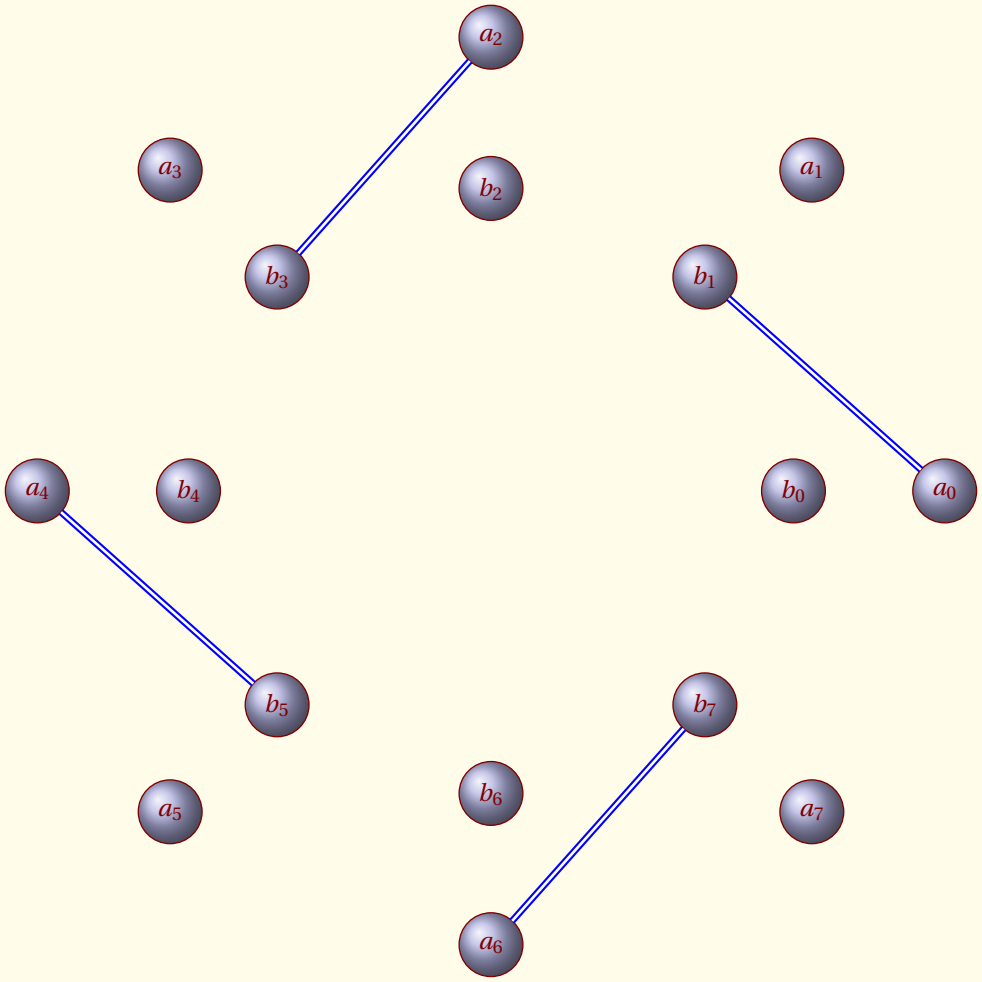

```
\begin{tikzpicture}
 \GraphInit[vstyle=Shade]
 \grEmptyCycle[prefix=a,RA=6]{9}
 \grEmptyCycle[prefix=b,RA=3]{9}
 \EdgeMod{a}{b}{9}{1}
\end{tikzpicture}
```

3.15 Edges between two graphs \EdgeMod*

\EdgeMod*{⟨*prefix1*⟩}{⟨*prefix2*⟩}{⟨*order*⟩}{⟨*step1*⟩}{⟨*step2*⟩}

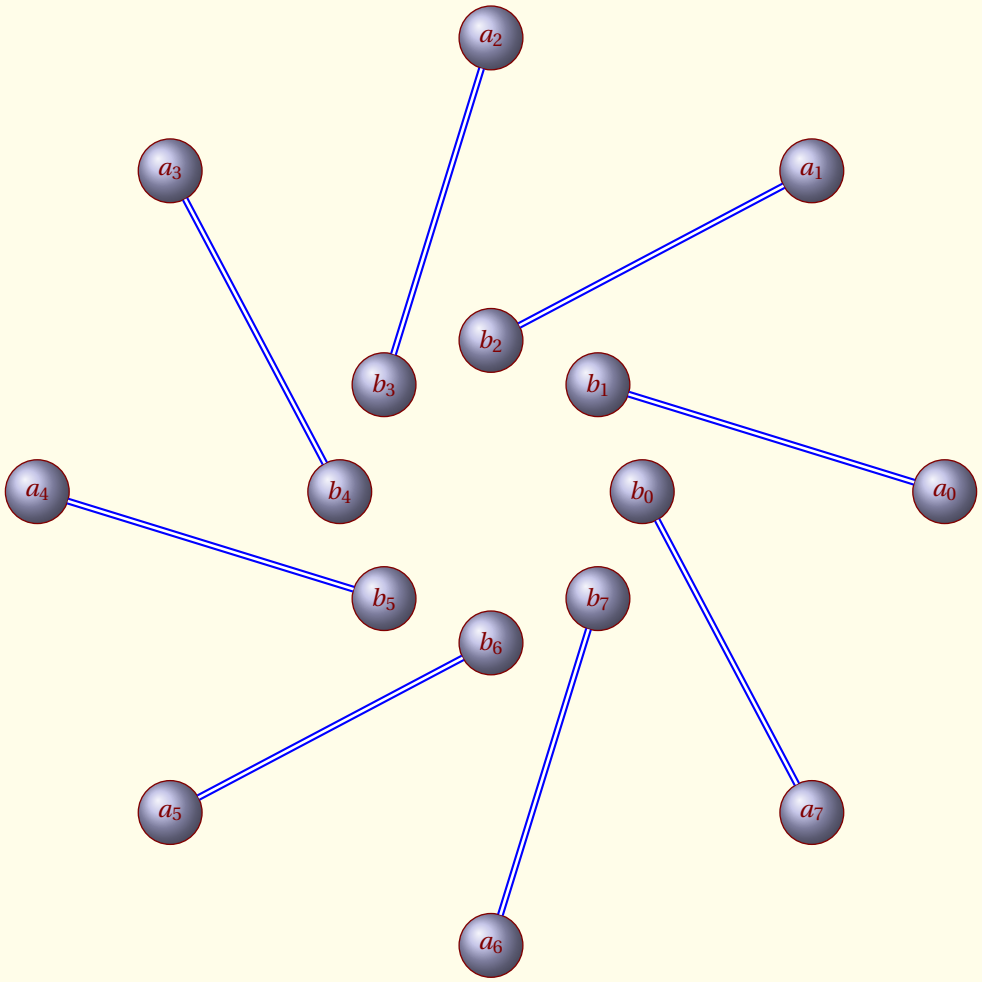
This macro works on two graphs with the same order. We get edges between v_i and v_j with i in $0,...,(\#3-1)$ with a step #5 and j = Mod(i+#4,#3).
#3 = `order` , *#4* = `step1` *and #5* = `step2`.

3.15.1 \EdgeMod*




```
\begin{tikzpicture}
 \GraphInit[vstyle=Shade]
 \grEmptyCycle[prefix=a,RA=6]{8}
 \grEmptyCycle[prefix=b,RA=4]{8}
 \EdgeMod*{a}{b}{8}{1}{2}
\end{tikzpicture}
```

3.15.2 EdgeMod*




```
\begin{tikzpicture}
 \GraphInit[vstyle=Shade]
 \grEmptyCycle[prefix=a,RA=6]{8}
 \grEmptyCycle[prefix=b,RA=2]{8}
 \EdgeMod*{a}{b}{8}{1}{1}
\end{tikzpicture}
```

3.16 Edges between two graphs \EdgeDoubleMod

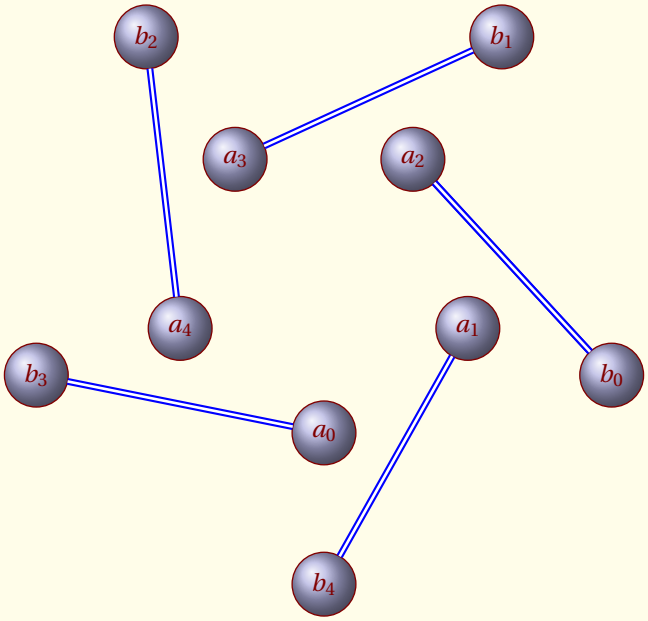
\EdgeDoubleMod{⟨*prefix1*⟩}{⟨*nb*⟩}{⟨*nb*⟩}{⟨*nb*⟩}{⟨*prefix2*⟩}{⟨*nb*⟩}{⟨*nb*⟩}{⟨*nb*⟩}{⟨*end*⟩}

For the first node, the numbers are : {⟨*order1*⟩}{⟨*start1*⟩}{⟨*add1*⟩}

For the second node, the numbers are : {⟨*order2*⟩}{⟨*start2*⟩}{⟨*add2*⟩}{⟨*end*⟩}

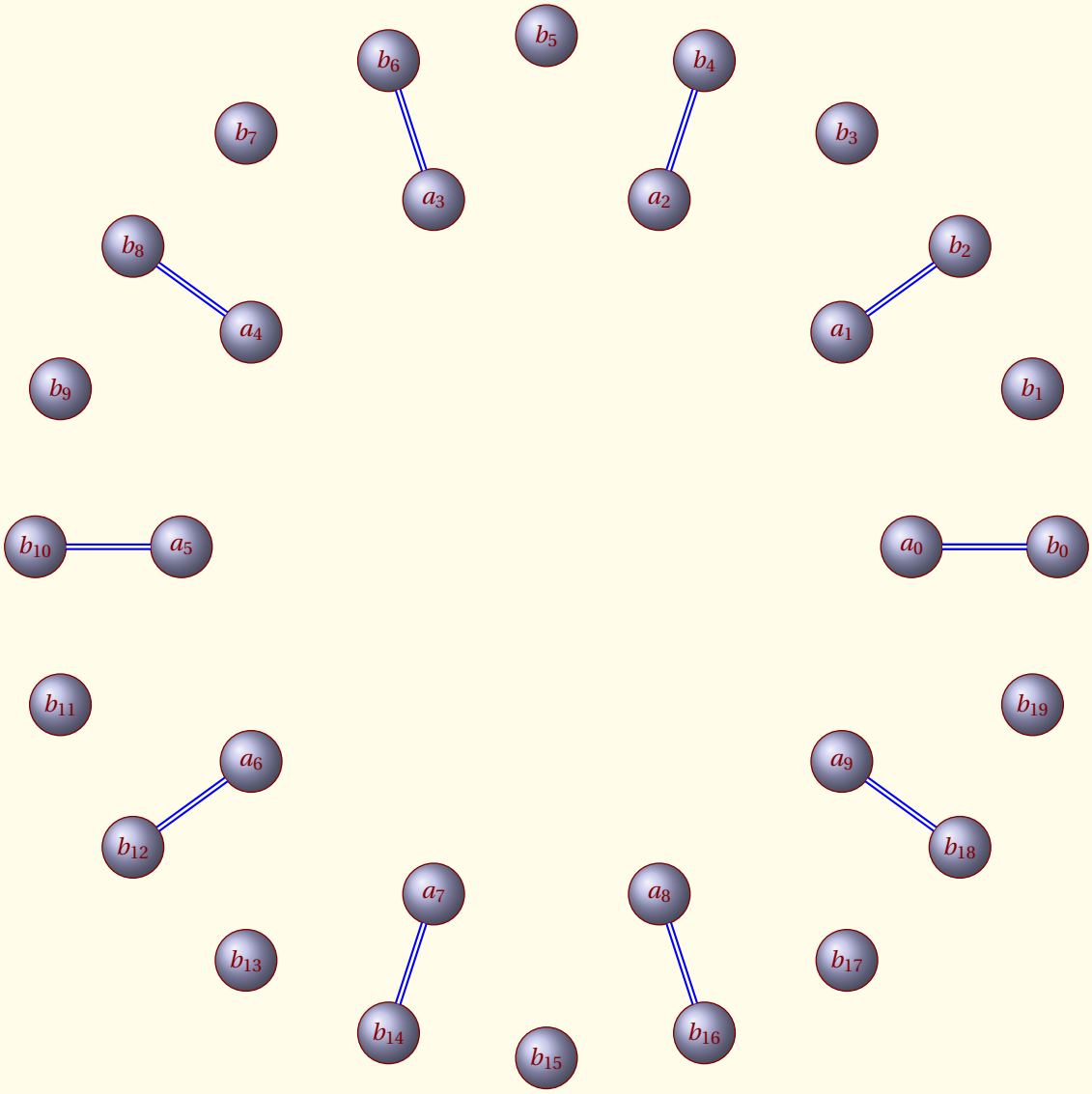
*Edges between v_i and v_j with i = Mod(#3+(#4*k),#2) and j=Mod(#7+(#8*k),#6) k is an integer from* 0 *to* end.
#2 = order1, *#3* = start1 *and #4* = add1.
#6 = order2, *#7* = start2 *and #8* = add2.

3.16.1 EdgeDoubleMod




```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
    \begin{scope}[rotate=-90]
      \grEmptyCycle[RA=2,prefix=a]{5}{2}
    \end{scope}
    \begin{scope}[rotate=-18]
      \grEmptyCycle[RA=4,prefix=b]{5}{2}
    \end{scope}
    \EdgeDoubleMod{b}{5}{0}{1}%
                  {a}{5}{2}{1}{5}
\end{tikzpicture}
```

3.16.2 EdgeDoubleMod with two graphs and different orders




```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \grEmptyCycle[prefix=a,RA=5]{10}
  \grEmptyCycle[prefix=b,RA=7]{20}
   \EdgeDoubleMod{a}{10}{0}{1}%
                 {b}{20}{0}{2}{10}
\end{tikzpicture}
```

SECTION 4

Classic Graphs

4.0.3 Cycle graph

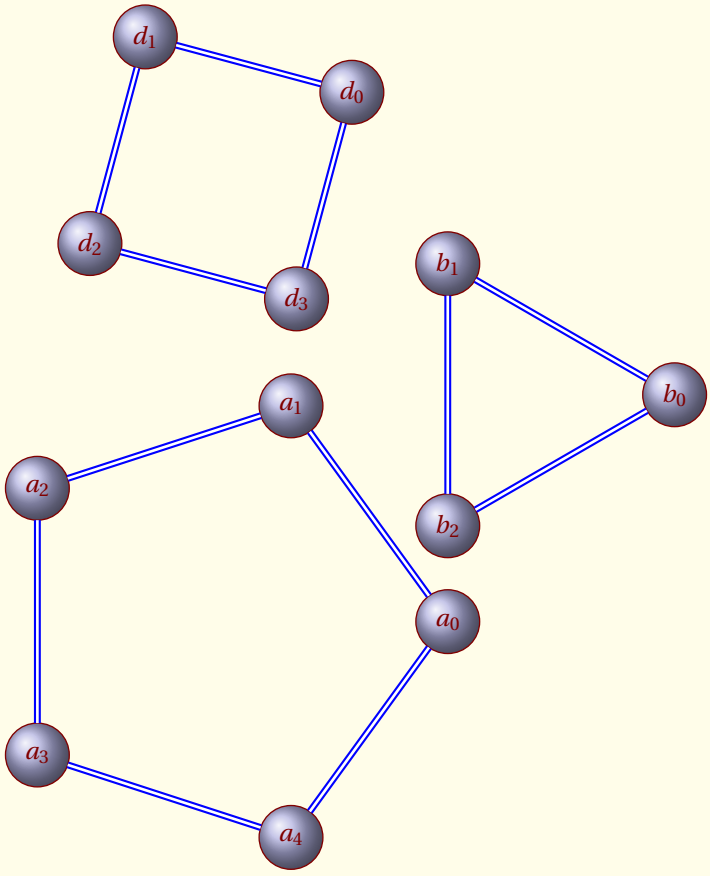
`\grCycle[⟨local options⟩]{⟨order⟩}`

A cycle graph C_n is a graph on n nodes containing a single cycle through all nodes. Cycle graphs can be generated using **`\grCycle`** *in the* **`tkz-berge.sty`** *package. Special cases include the triangle graph and the square graph.*

External links :

- MathWorld - CycleGraph by E.Weisstein
- Wikipedia

4.0.4 Special cases : the triangle graph and the square graph




```
\begin{tikzpicture}
    \GraphInit[vstyle=Shade]
    \grCycle[prefix=a,RA=3]{5}
    \grCycle[x=4,y=3,prefix=b,RA=2]{3}
    \grCycle[prefix=d,y=6,rotation=30,RA=2]{4}
\end{tikzpicture}
```

4.0.5 Complete graph

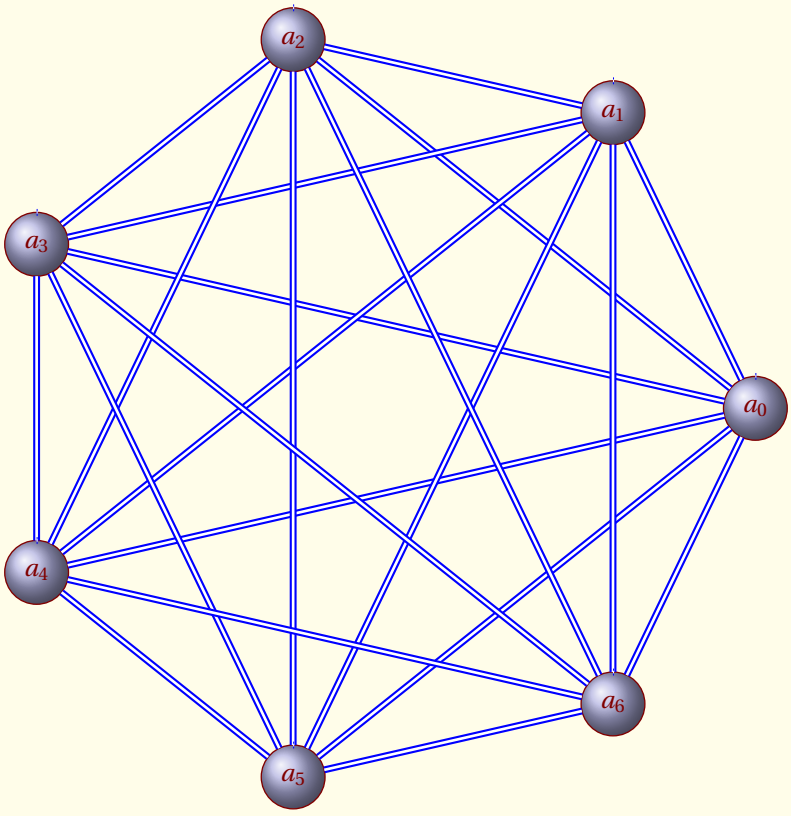
`\grComplete[⟨local options⟩]{⟨order⟩}`

The more simple definition is "an undirected graph with an edge between every pair of vertices" or a complete graph is a simple graph in which each pair of graph vertices is connected by an edge. The complete graph with n graph vertices is denoted K_n*. This graph has* $\frac{n(n-1)}{2}$ *undirected edges.*
Geometrically, K_3 *relates to a triangle,*K_4 *a tetrahedron is the tetrahedral graph as well as the wheel graph ,* K_5 *a pentachoron, etc . . .*

External links :

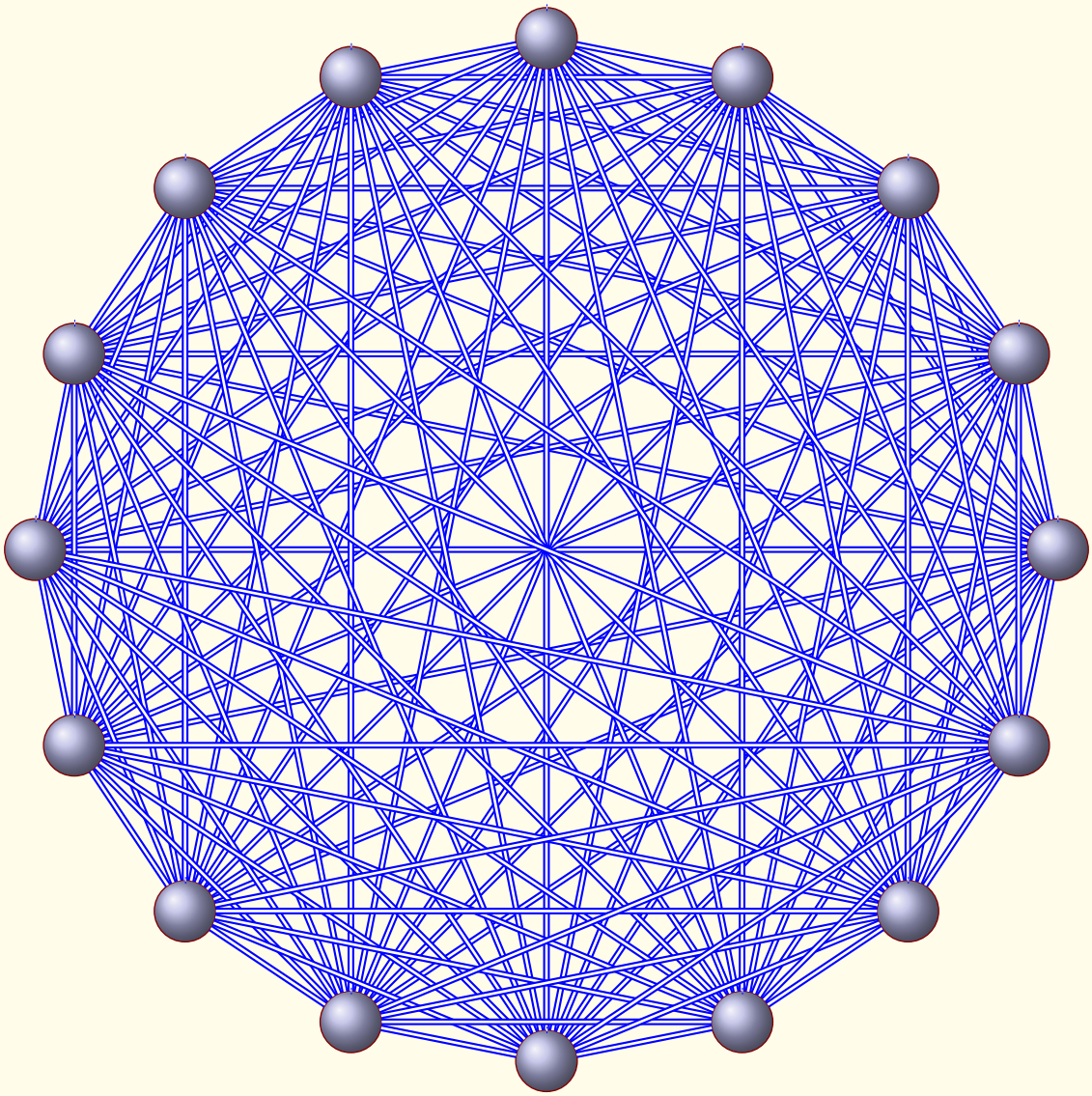
- Wikipedia
- MathWorld - Complete graph by E.Weisstein

4.0.6 Complete Graph order 4




```
\begin{tikzpicture}
  \renewcommand*{\VertexBallColor}{green!50!black}
  \GraphInit[vstyle=Shade]
  \grComplete[RA=5]{7}
\end{tikzpicture}
```

4.0.7 Complete Graph order 4



```
\begin{tikzpicture}
   \renewcommand*{\VertexBallColor}{green!50!black}
   \GraphInit[vstyle=Shade]
   \SetVertexNoLabel
   \grComplete[RA=7]{16}
\end{tikzpicture}
```

4.0.8 Circulant graph

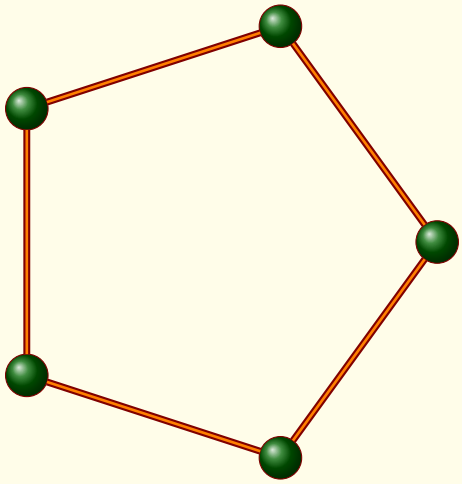
`\grCirculant[⟨local options⟩]{⟨order⟩}`

The circulant graph is defined for any order n at least 3, and every subset L *of integers which are less than or equal to* $n/2$*. A circulant graph is a graph in which the* i*th graph vertex is adjacent to the* $(i+j)$*th and* $(i-j)$*th graph vertices for each* j *in a list* L *. The circulant graphs with* $L=\{1;\ldots;[n/2]\}$ *gives the complete graphs and the circulant graph with* $L=\{1\}$ *gives the cyclic graphs. The Möbius ladders are examples of circulant graphs.*
In graph theory, a graph whose adjacency matrix is circulant is called a circulant graph.
The circulant graph on vertices on a list of nodes is implemented as **`\grCirculant`** *in the* **`tkz-berge.sty`** *package.*

External links :
MathWorld - CirculantGraph by E.Weisstein

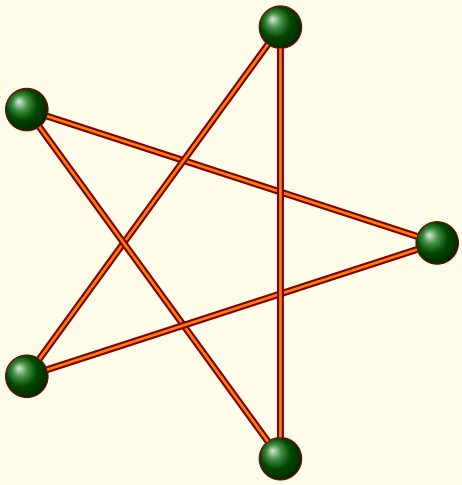
4.0.9 Graph order 5 with L={1}

This is a cycle graph.



```
\begin{tikzpicture}
   \grCirculant[RA=3]{5}{1}%
\end{tikzpicture}
```

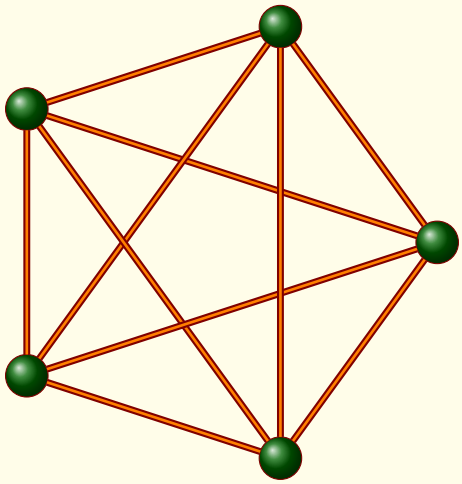
4.0.10 Graph order 5 with L={2}



```
\begin{tikzpicture}
   \grCirculant[RA=3]{5}{2}%
\end{tikzpicture}
```

4.0.11 Graph order 5 with L={1,2}

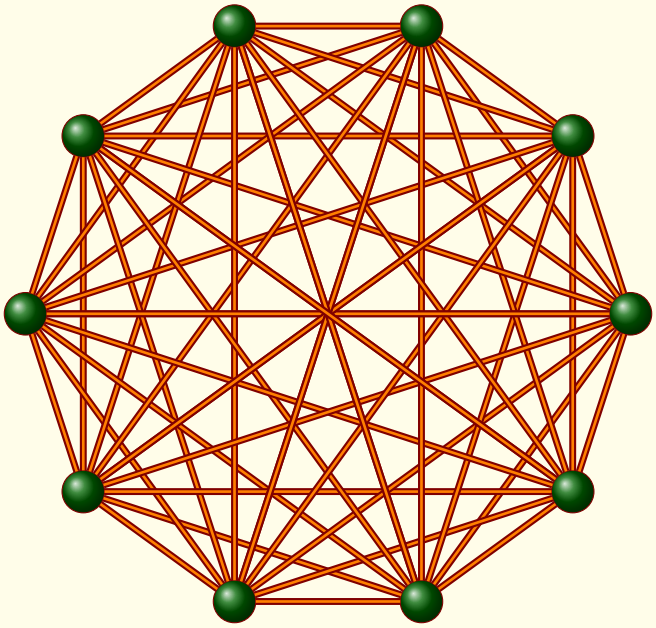
This graph is complete with an order 5.



```
\begin{tikzpicture}
   \grCirculant[RA=3]{5}{1,2}%
\end{tikzpicture}
```

4.0.12 Graph order 10 with L={1,2,3,4,5}

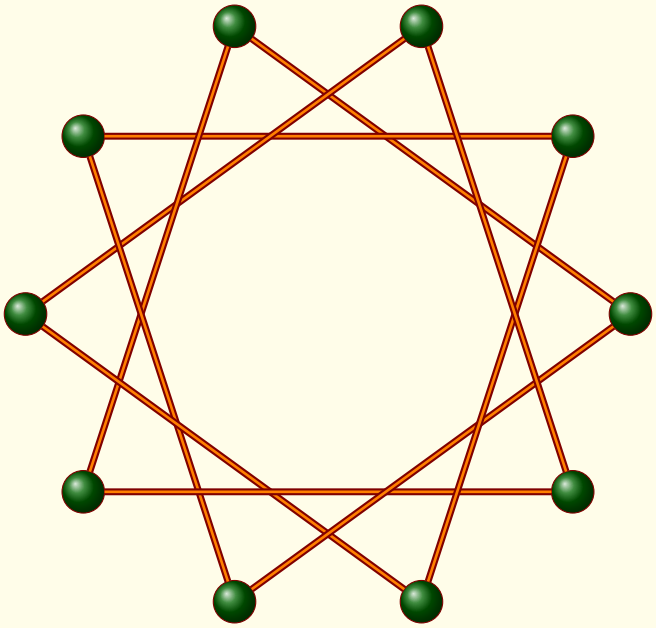
This graph is also complete



```
\begin{tikzpicture}
    \grCirculant[RA=4]{10}{1,2,3,4,5}%
\end{tikzpicture}
```

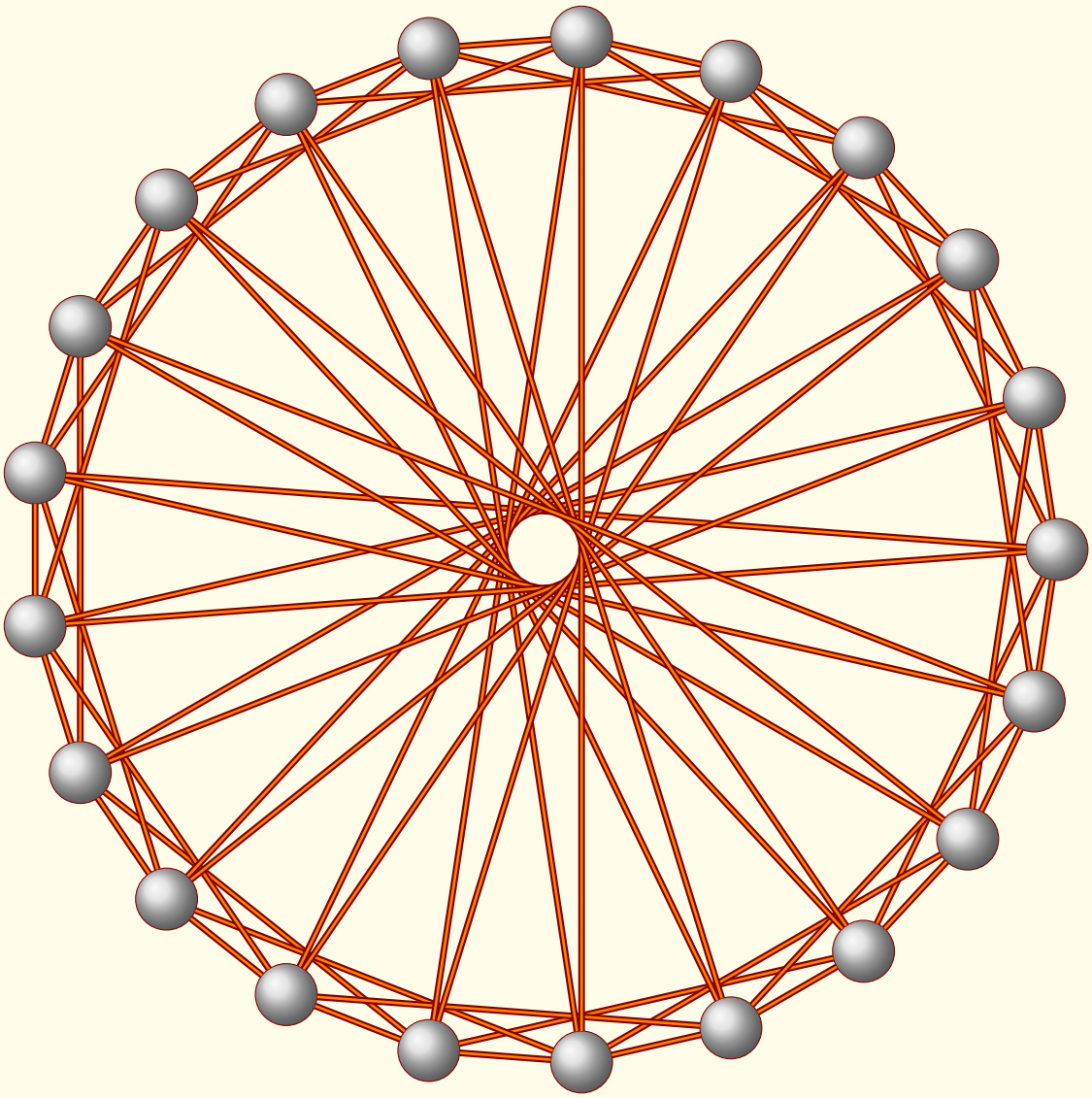
It's interesting to remark that the numbers 3 and 10 are primer, so if L = {3} the graph is containing an Eulerian circuit.

4.0.13 Graph order 10 with L={3}



```
\begin{tikzpicture}
    \grCirculant[RA=4]{10}{3}%
\end{tikzpicture}
```

4.0.14 Graph order 21 with L={1,3,10}



4.0.15 Star graph

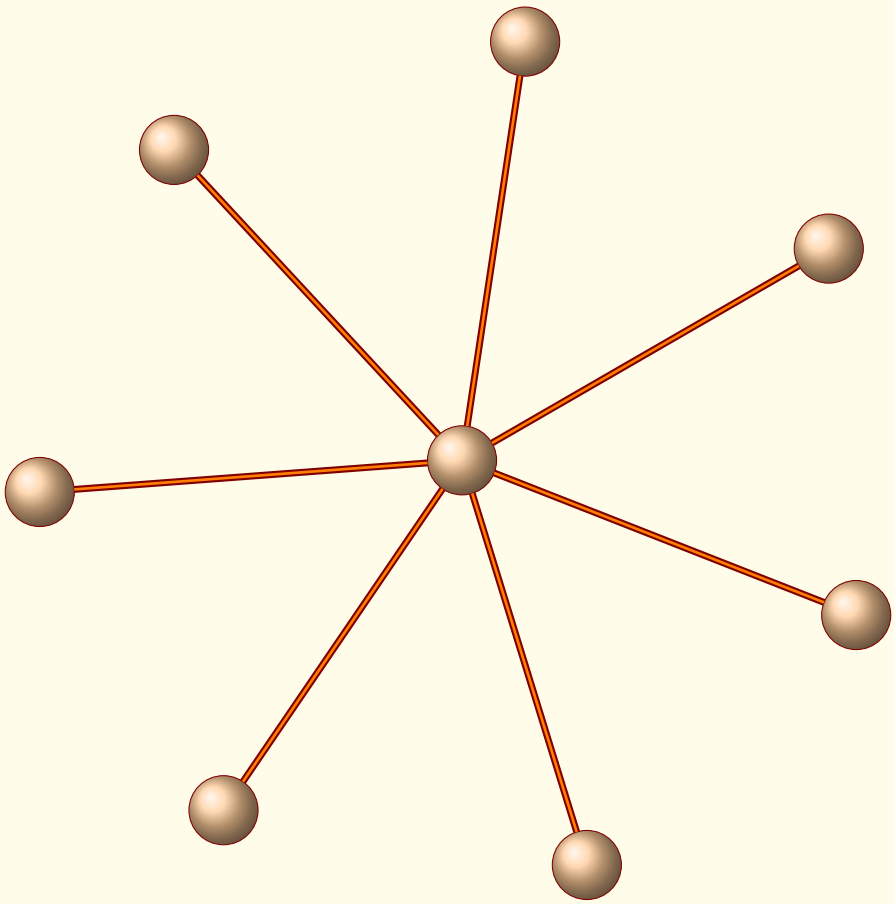
`\grStar[⟨local options⟩]{⟨order⟩}`

A star graph S_n *is a n-graph with one node having vertex degree* $n-1$ *and the other* $n-1$ *having vertex degree* 1. *Star graphs can be generated using* **`\grStar`** *in the* **`tkz-berge.sty`** *package.*

External links :

- MathWorld - StarGraph by Weisstein

4.0.16 Star graph



```
\begin{tikzpicture}[rotate=30,scale=.8]
  \grStar[RA=7]{8}%
\end{tikzpicture}
```

4.0.17 Square graph

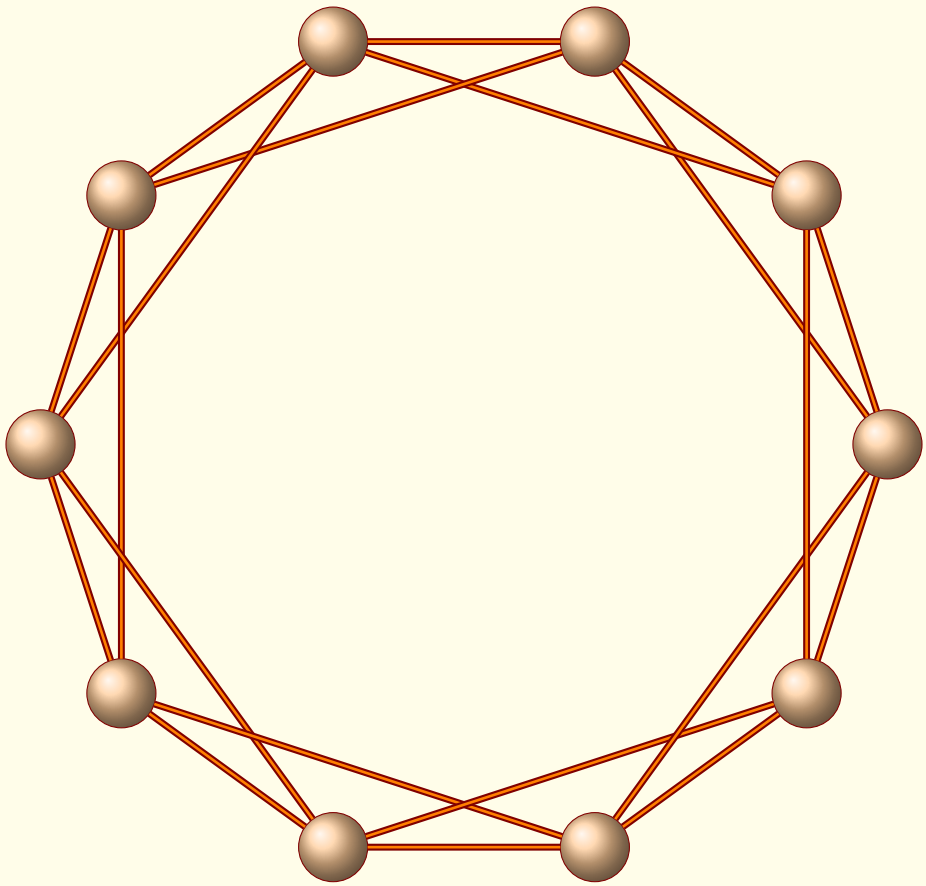
`\grSQCycle[⟨local options⟩]{⟨Number⟩}`

A star graph S_n *is a n-graph with one node having vertex degree* $n-1$ *and the other* $n-1$ *having vertex degree* 1. *Star graphs can be generated using* **`\grStar`** *in the* **`tkz-berge.sty`** *package.*

External links :

- MathWorld - SquareGraph by Weisstein

4.0.18 Square Cycle graph



```
\begin{tikzpicture}[scale=.8]
  \grSQCycle[RA=7]{10}%
\end{tikzpicture}
```

4.0.19 Wheel graph

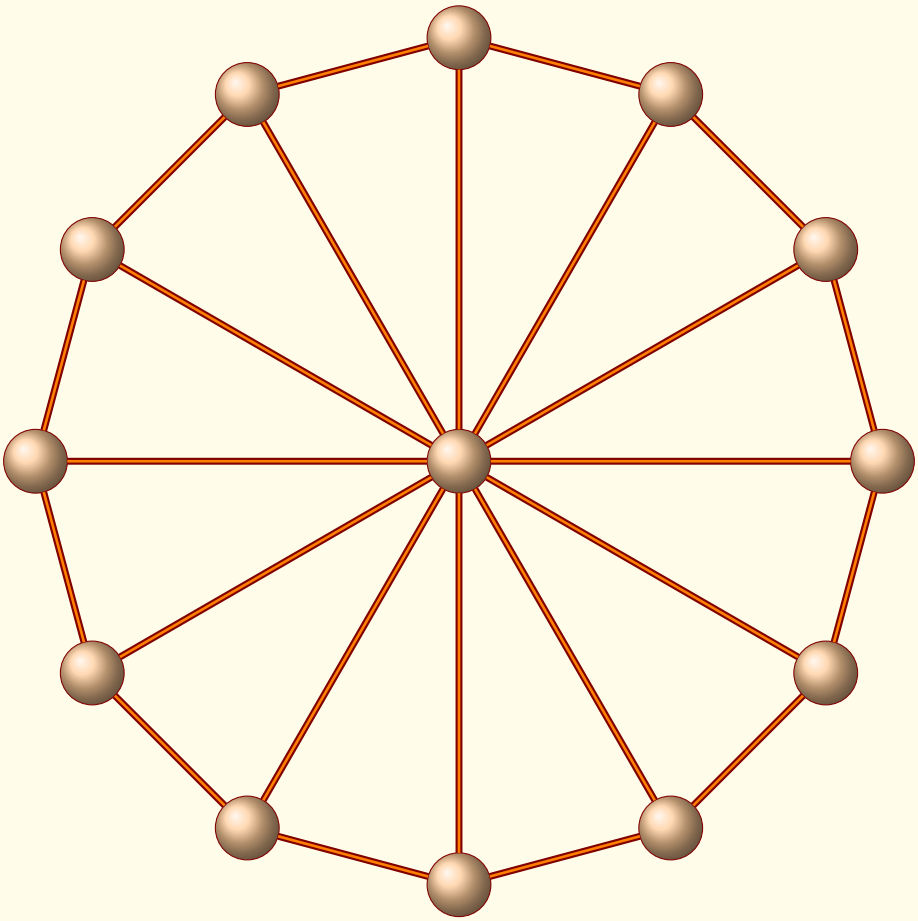
`\grWheel[⟨local options⟩]{⟨Number⟩}`

A wheel graph of order n is a graph that contains a cycle of order $n-1$, and for which every vertex in the cycle is connected to one other vertex. The wheel can be defined as the graph , where is the singleton graph and is the cycle graph.

External links :

- MathWorld - WheelGraph by Weisstein

4.0.20 Wheel graph



```
\begin{tikzpicture}[scale=.8]
      \grWheel[RA=7]{13}%
\end{tikzpicture}
```

4.0.21 Ladder graph

`\grLadder[⟨local options⟩]{⟨Number⟩}`

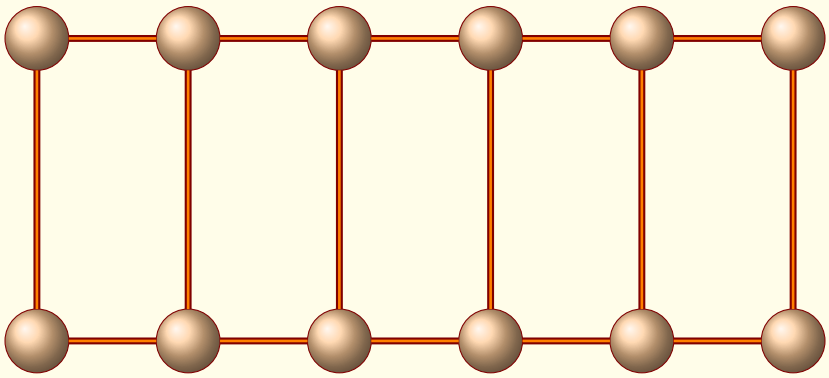
options	default	definition
RA	4	radius circle n°1
RS	0	distance between two lines
prefix	a	prefix for vertices
prefixx	b	prefix for vertices
Math	false	math mode

The ladder graph L_n *or cyclic ladder graph is equivalent to the grid graph having two rails and n rungs between them.*

External links :

- MathWorld - LadderGraph by Weisstein

4.0.22 Ladder graph



```
\begin{tikzpicture}
      \grLadder[RA=2,RS=4]{6}%
\end{tikzpicture}
```

4.0.23 Prism graph

`\grPrism[⟨local options⟩]{⟨Number⟩}`

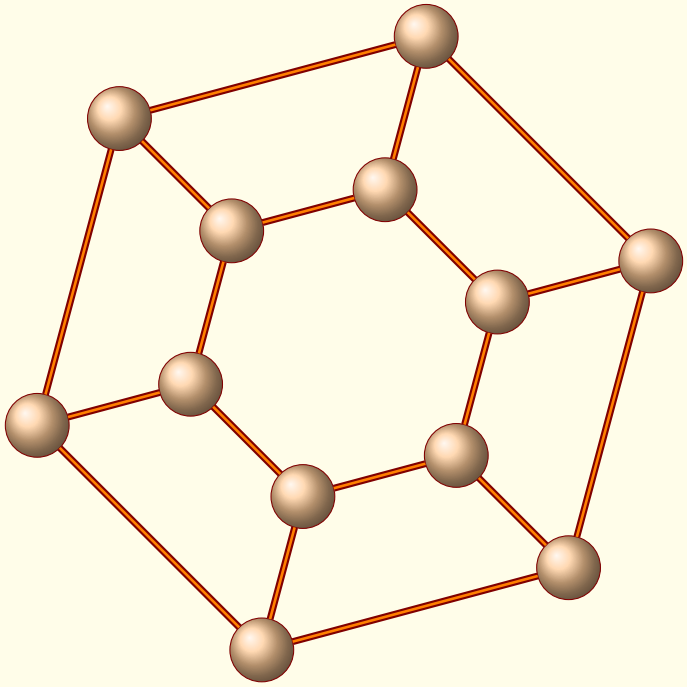
options	default	definition
RA	4	radius circle n°1
RB	3	radius circle n°2
prefix	a	prefix for vertices
prefixx	b	prefix for vertices
Math	false	math mode

An n-prism graph has $2n$ nodes and $3n$ edges, and is equivalent to the generalized Petersen graph with arguments n and 1. For odd n, the n-prism is isomorphic to the circulant graph with an order $2n$ and with arguments 2 and n. The 3-prism graph is the line graph of the complete bipartite graph with arguments 2 and 3 . The 4-prism graph is isomorphic with the cubical graph.

External links :

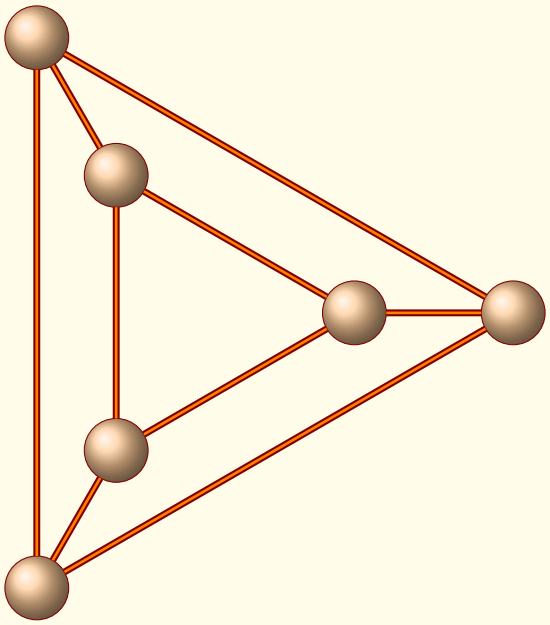
- MathWorld - Prism Graph by Weisstein

4.0.24 Cycle Ladder graph



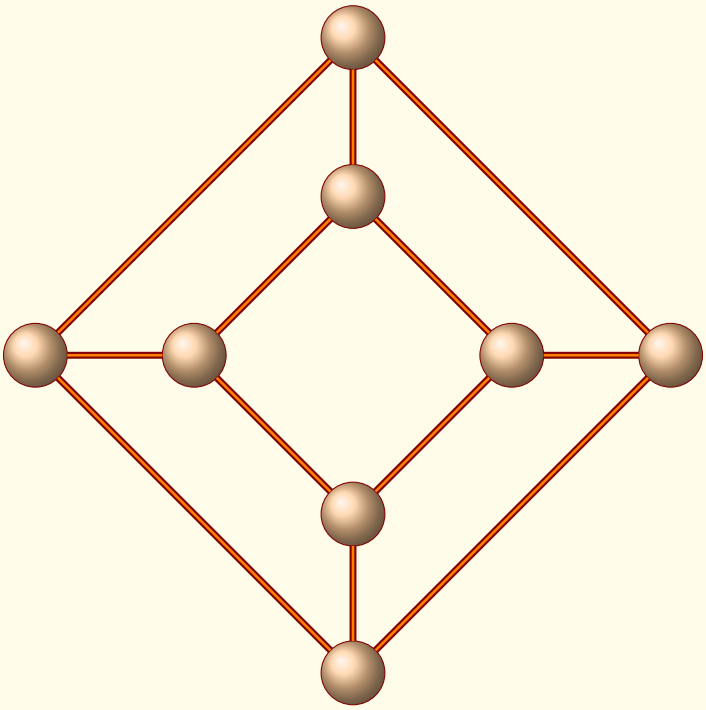
```
\begin{tikzpicture}[rotate=15,scale=.7]
  \grPrism[RA=6,RB=3]{6}%
\end{tikzpicture}
```

4.0.25 Cycle Ladder graph number 3



```
\begin{tikzpicture}[scale=.7]
  \grPrism[RA=6,RB=3]{3}%
\end{tikzpicture}
```

4.0.26 Cycle Ladder graph number 4



```
\begin{tikzpicture}[scale=.7]
  \grPrism[RA=6,RB=3]{4}%
\end{tikzpicture}
```

4.0.27 Complete Bipartite graph

`\grCompleteBipartite[⟨local options⟩]{⟨Number 1⟩}{⟨Number 2⟩}`

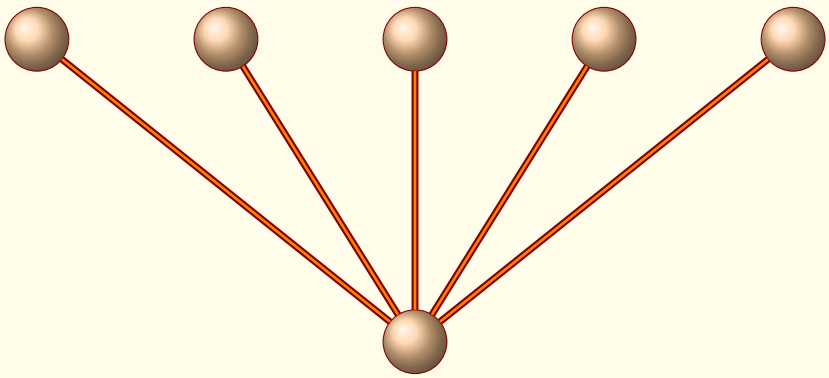
options	default	definition
RA	4	radius circle n°1
RB	3	radius circle n°2
RS	1	distance between two lines
form	1	integer to obtain a new embedding of a graph
prefix	a	prefix for vertices
prefixx	b	prefix for vertices
Math	false	math mode

A complete bipartite graph is a bipartite graph (i.e., a set of graph vertices decomposed into two disjoint sets such that no two graph vertices within the same set are adjacent) such that every pair of graph vertices in the two sets are adjacent.

External links :

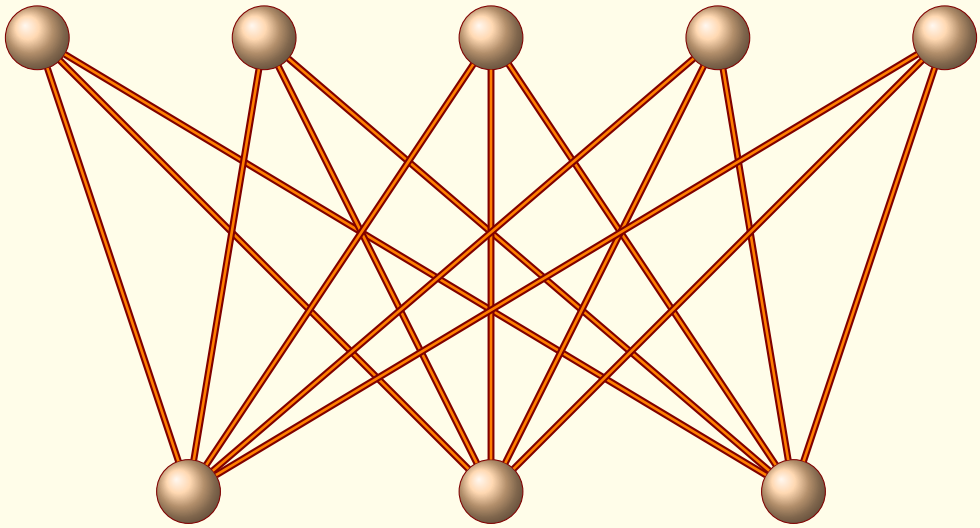
- MathWorld - CompleteBipartite Graph by Weisstein

4.0.28 Bipartite graph 1,5



```
\begin{tikzpicture}
   \grCompleteBipartite[RA=4,RB=2.5,RS=4]{1}{5}
\end{tikzpicture}
```

4.0.29 Bipartite graph 3,5



```
\begin{tikzpicture}
   \grCompleteBipartite[RA=4,RB=3,RS=6]{3}{5}
\end{tikzpicture}
```

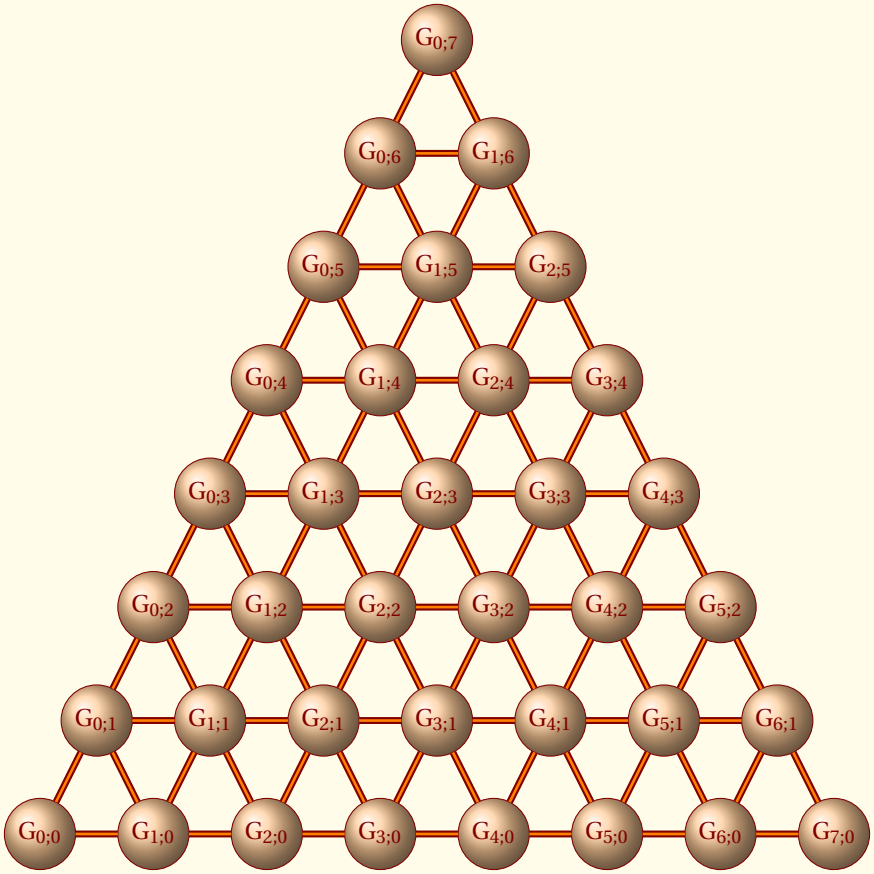
4.0.30 Triangular Grid graph

`\grTriangularGrid[⟨local options⟩]{⟨Number⟩}`

options	default	definition
RA	4	distance between two vertices
form	1	integer to obtain a new embedding of a graph
prefix	a	prefix for vertices
Math	false	math mode

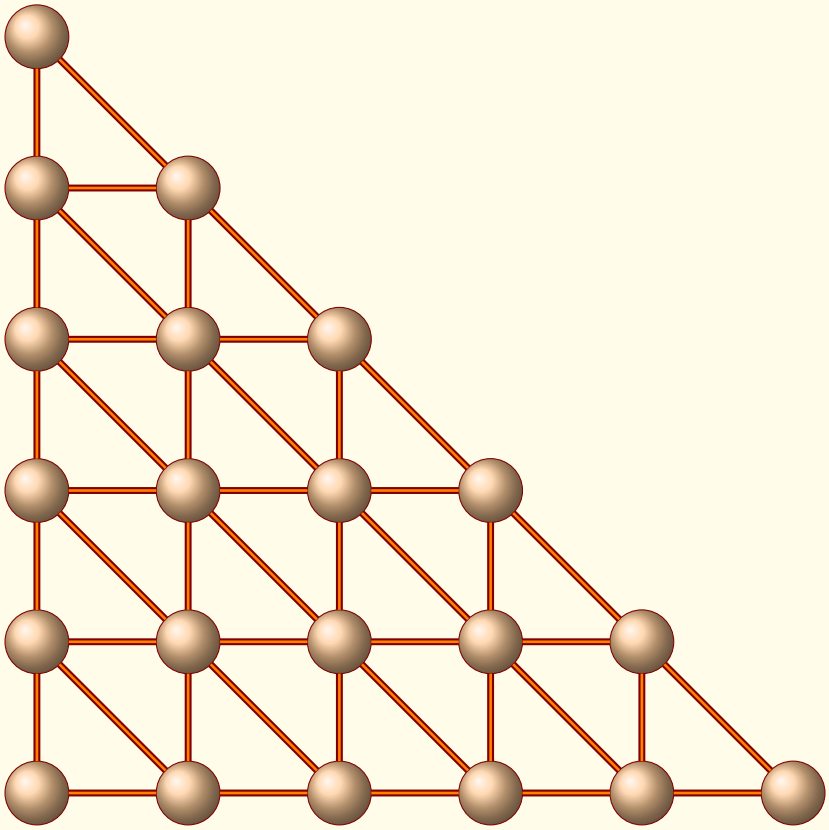
Number=*n is the number of vertices of the first row then the graph order is* $\frac{n(n-1)}{2}$*. There are three embeddings. You can use the option* **form** *with an integer between* 1 *and* 3*.*

4.0.31 n=8 order=28 form 1



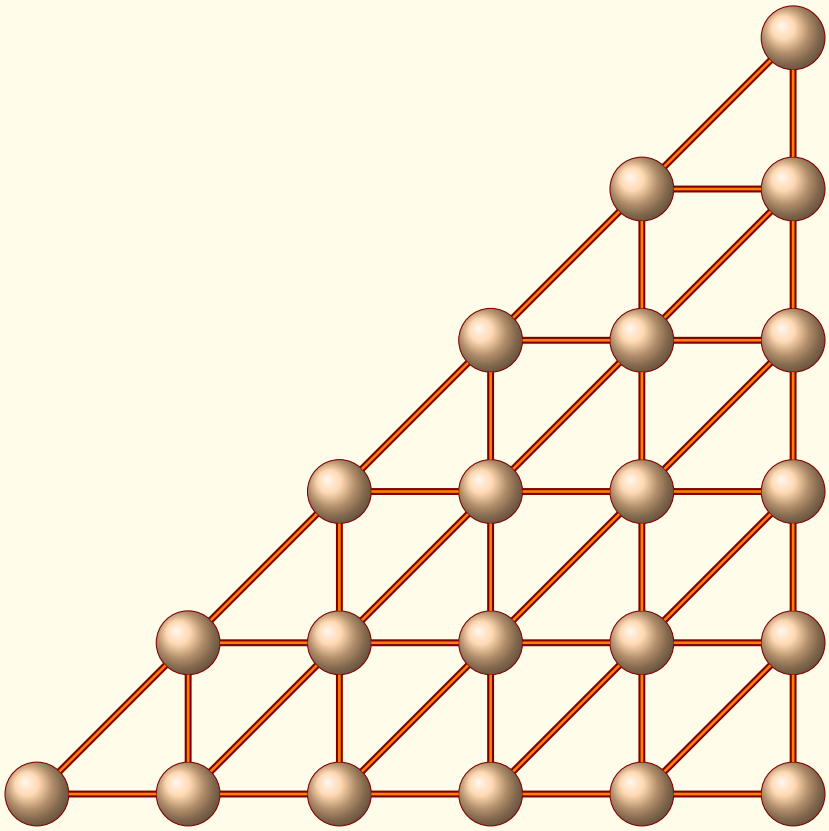

```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \SetVertexLabel
  \grTriangularGrid[prefix=G,Math,RA=1.5]{8}%
\end{tikzpicture}
```

4.0.32 n=6 order=15 form 2



```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \SetVertexNoLabel
  \grTriangularGrid[RA=2,form=2]{6}%
\end{tikzpicture}
```

4.0.33 n=6 order=15 form 3



```
\begin{tikzpicture}
  \GraphInit[vstyle=Shade]
  \SetVertexNoLabel
  \grTriangularGrid[RA=2,form=3]{6}%
\end{tikzpicture}
```

4.0.34 LCF Lederberg-Coxeter-Fruchte

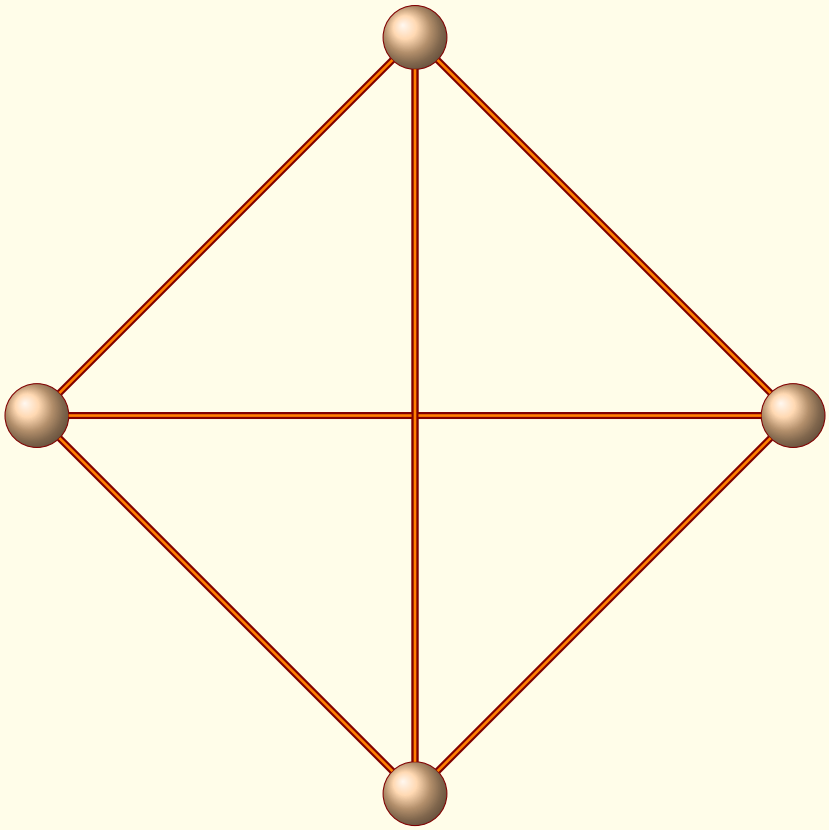
`\grLCF[⟨RA=⟨Number⟩⟩] {⟨List of numbers⟩} {⟨Number⟩}`

LCF = Lederberg-Coxeter-Fruchte (see the link below for some examples).

External links :

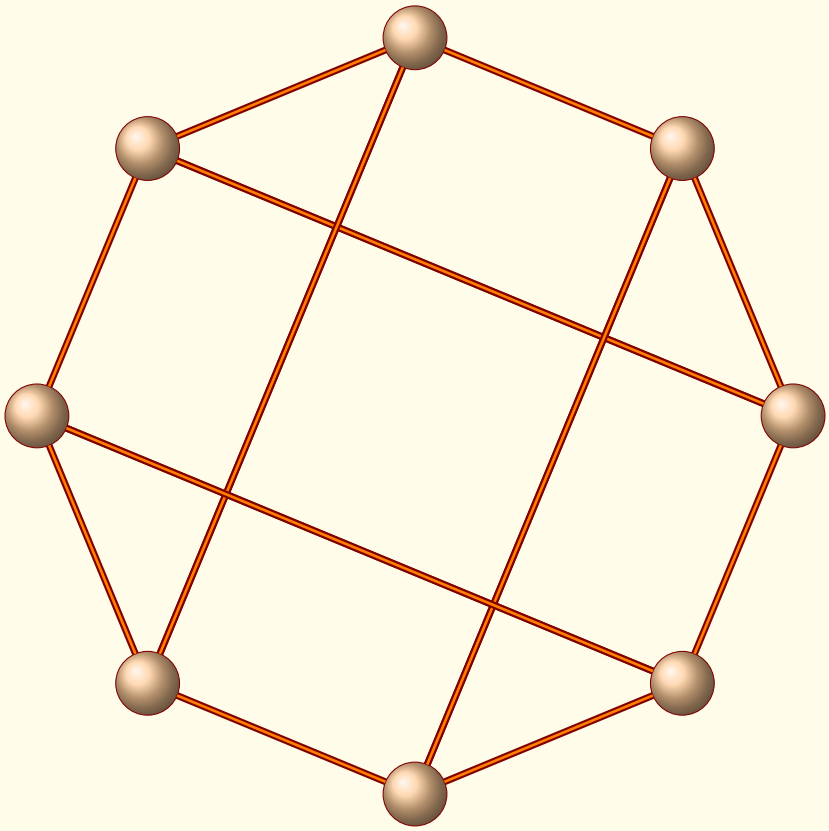
- MathWorld-LCF Notation by Weisstein

4.0.35 $[2,-2]^2$



```
\begin{tikzpicture}%
       \grLCF[RA=5]{2,-2}{2}%
\end{tikzpicture}
```

4.0.36 $[3,-3]^4$

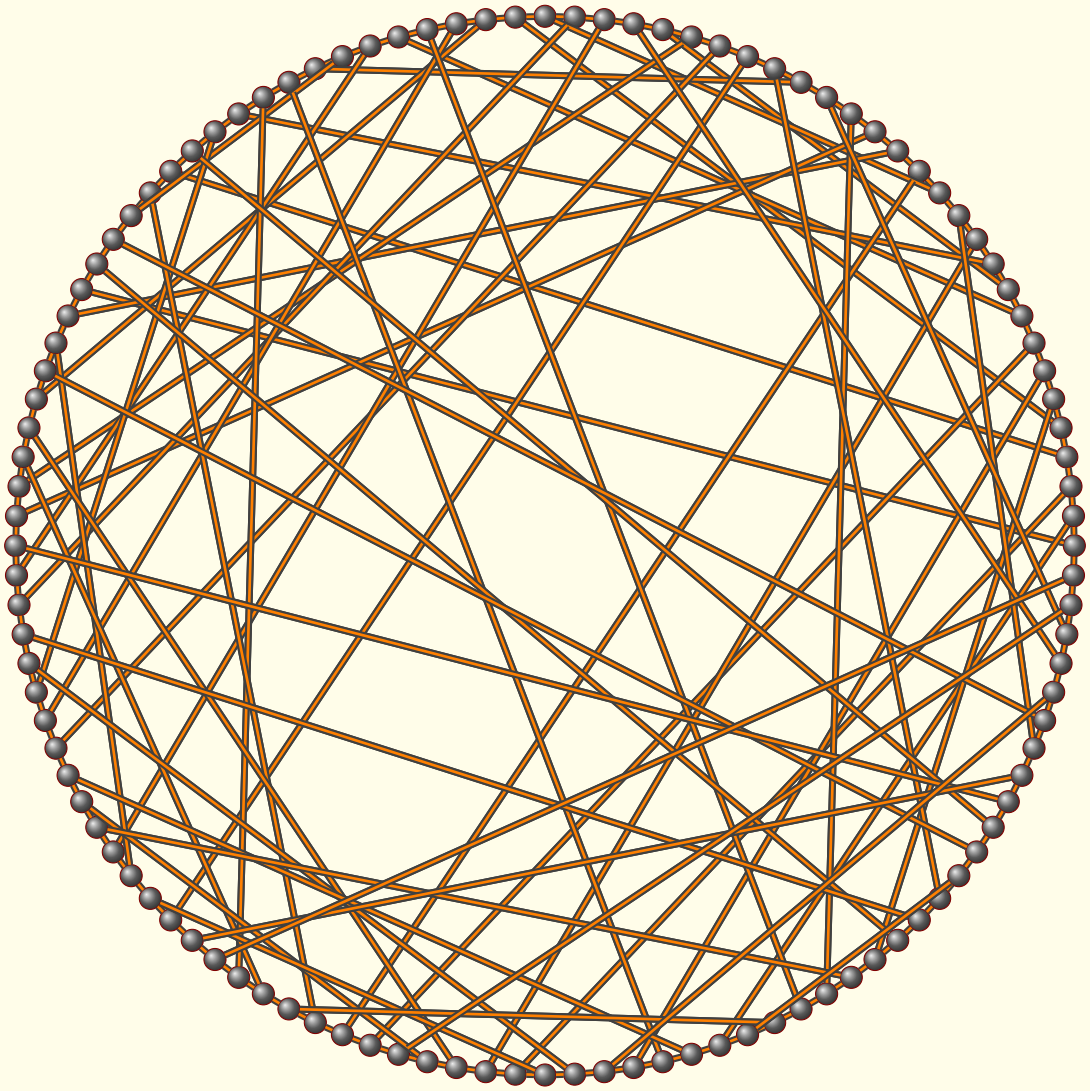


```
\begin{tikzpicture}%
       \grLCF[RA=5]{3,-3}{4}%
\end{tikzpicture}
```

4.0.37 Ljubljana graph

From Wikipedia http://en.wikipedia.org/wiki/Ljubljana_graph
The Ljubljana graph was first published in 1993 by Brouwer, Dejter and Thomassen. In 1972, Bouwer was already talking of a 112-vertices edge- but not vertex-transitive cubic graph found by R. M. Foster, but unpublished. Conder, Malnič, Marušič, Pisanski and Potočnik rediscovered this 112-vertices graph in 2002 and named it the Ljubljana graph after the capital of Slovenia. They proved that it was the unique 112-vertices edge- but not vertex-transitive cubic graph and therefore that was the graph found by Foster.
It can be represented in LCF notation as :

$$\Big[47,-23,-31,39,25,-21,-31,-41,25,15,29,-41,-19,15,-49,33,39,-35,-21,17,\\ -33,49,41,31,-15,-29,41,31,-15,-25,21,31,-51,-25,23,9,-17,51,35,-29,21,\\ -51,-39,33,-9,-51,51,-47,-33,19,51,-21,29,21,-31,-39\Big]^2$$



```
\GraphInit[vstyle=Art]
\SetGraphArtColor{black!50}{darkgray}
\tikzset{VertexStyle/.append style = {
                              minimum size         =  3pt}}
\begin{tikzpicture}%
 \grLCF[RA=7]{47, -23, -31, 39, 25, -21, -31, -41, 25, 15, 29, -41, -19, 15,%
  -49, 33, 39, -35, -21, 17, -33, 49, 41, 31, -15, -29, 41, 31, -15, -25, 21,%
   31, -51, -25, 23, 9, -17, 51, 35, -29, 21, -51, -39, 33, -9, -51, 51, -47,%
    -33, 19, 51, -21, 29, 21, -31, -39}{2}%
\end{tikzpicture}
```

SECTION 5

Macros and Styles

5.1 How to change the background color and text color

You can use the following macro :

\tkzSetUpColors[⟨*local options*⟩]

Options	default	definition
background	white	couleur du fond
text	black	couleur du texte

5.2 Modification of labels \AssignVertexLabel

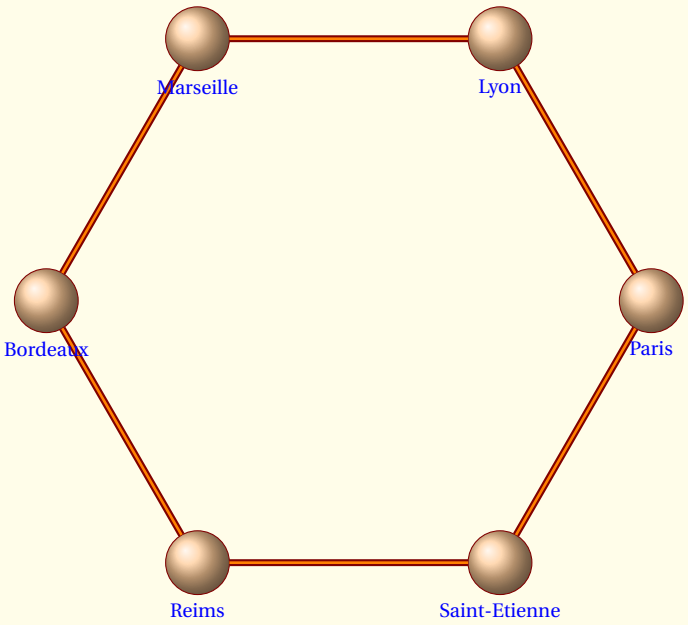
\AssignVertexLabel[⟨*local options*⟩]{⟨*prefix*⟩}{⟨*List of names*⟩}

Arguments	example
prefix	**\AssignVertexLabel{a}{Alter}**
List of names	**\AssignVertexLabel{a}{Paris,Lyon}**

Options	default	definition
size	**\normalsize**	taille de la fonte
color	black	couleur du texte
Math	false	math mode

5.2.1 AssignStyle and \AssignVertexLabel

First step : We create an empty graph without labels.
Second step : We place labels with the macro **\AssignVertexLabel**




```
\begin{tikzpicture}
  \SetVertexNoLabel
  \grCycle{6}
  \tikzset{AssignStyle/.append style = {below=12pt}}
  \AssignVertexLabel[color = blue,%
                     size  = \footnotesize]{a}{%
   Paris,Lyon,Marseille,Bordeaux,Reims,Saint-Etienne}
\end{tikzpicture}
```

Index

G

X