Теория на графите

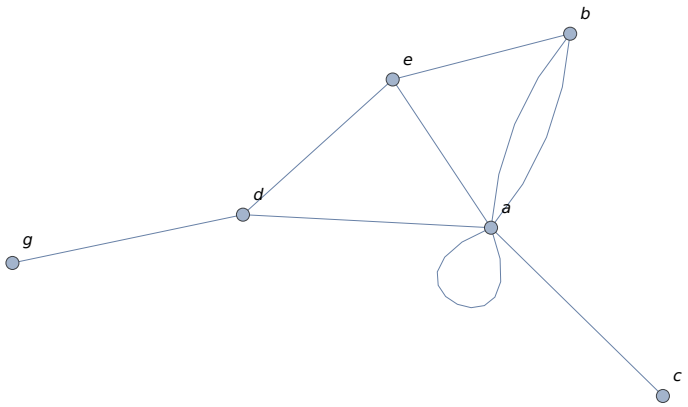
1. Изчертаване на ненасочен граф

(*Esc ue Esc*)

In[3]:=

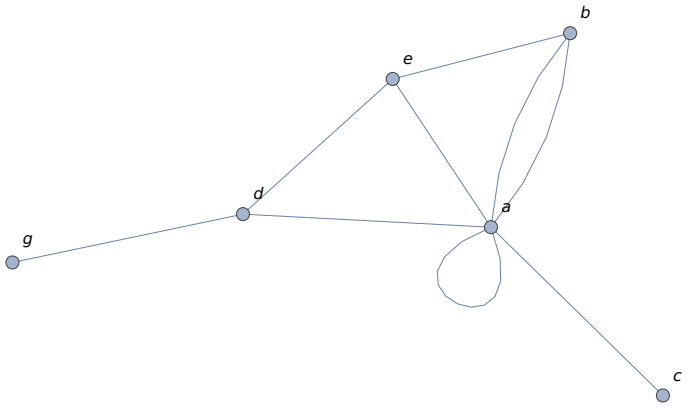
```
(* Първи начин *)
graph1 =Graph[{a,b,c,d,e,g},
{a↔a,a↔b,b↔a,a↔c,a↔e,a↔d,d↔g,b↔e,e↔d},VertexLabels →"Name"]
```

Out[3]=




```
(*Втори начин *)
graph2 =Graph[{a,b,c,d,e,g},{a→a,a→b,b→a,a→c,a→e,a→d,d→g,b→e,e→d},
VertexLabels →"Name",DirectedEdges →False]
```

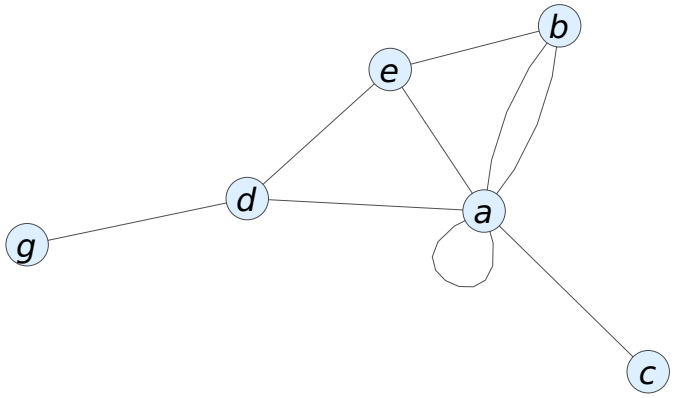
Out[50]=



Задаване на стил на изчертаване на граф

In[4]:=
```
graph22 =Graph[{a,b,c,d,e,g},{a↔a,a↔b,b↔a,a↔c,a↔e,a↔d,d↔g,b↔e,e↔d},
VertexLabels →Placed["Name",Center],VertexStyle →LightBlue ,VertexSize  → 0.25 ,
VertexLabelStyle  →  Directive[Black , Italic , 18],EdgeStyle →Black]
```

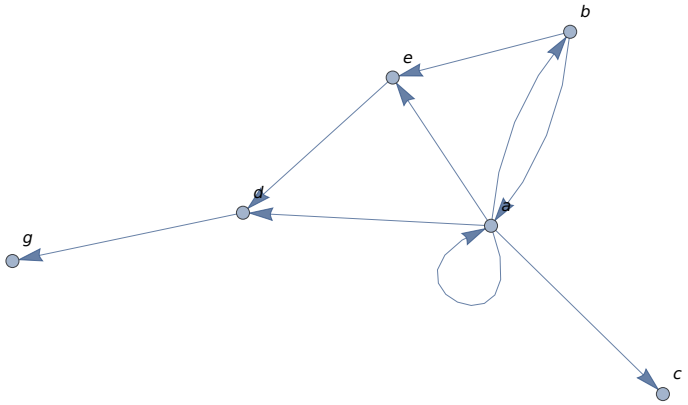
Out[4]=



2. Изчертаване на насочен граф

```
graph3 =Graph[{a,b,c,d,e,g},{a→a,a→b,b→a,a→c,a→e,a→d,d→g,b→e,e→d},
VertexLabels →"Name"]
```

Out[5]=



Извеждане на списък на върховете и ребрата

In[6]:=
```
VertexList [graph1 ]
VertexList [graph3 ]
```

Out[6]= {a, b, c, d, e, g}

Out[7]= {a, b, c, d, e, g}

In[8]:=
```
EdgeList [graph1 ]
EdgeList [graph3 ]
```

Out[8]= {a ↣ a, a ↣ b, b ↣ a, a ↣ c, a ↣ e, a ↣ d, d ↣ g, b ↣ e, e ↣ d}

Out[9]= {a ↣ a, a ↣ b, b ↣ a, a ↣ c, a ↣ e, a ↣ d, d ↣ g, b ↣ e, e ↣ d}

Извеждане на степените на върховете

In[10]:=
```
VertexDegree [graph1 ]
VertexInDegree [graph3 ]
VertexOutDegree [graph3 ]
```

Out[10]= {7, 3, 1, 3, 3, 1}

Out[11]= {2, 1, 1, 2, 2, 1}

Out[12]= {5, 2, 0, 1, 1, 0}

Извеждане на матрицата на съседство

In[13]:=
```
AdjacencyMatrix [graph1 ]//MatrixForm
AdjacencyMatrix [graph3 ]//MatrixForm
```

Out[13]//MatrixForm=

$$\begin{pmatrix} 1 & 2 & 1 & 1 & 1 & 0 \\ 2 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

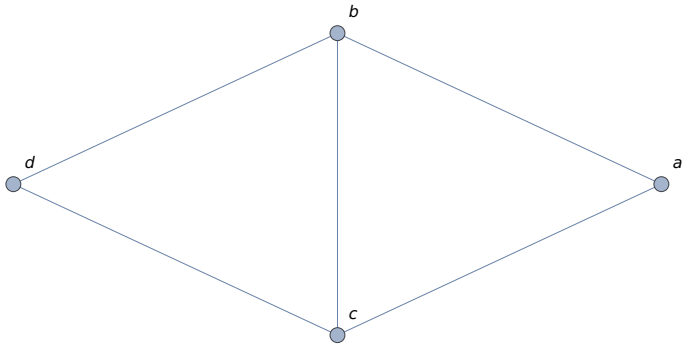
Out[14]//MatrixForm=

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Построяване на граф по матрицата на съседство

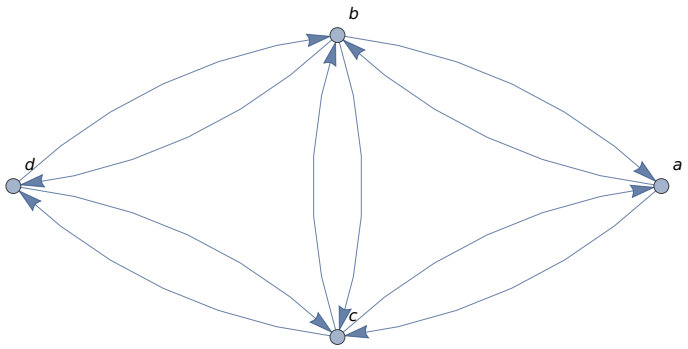
In[52]:=
```
graph4 =AdjacencyGraph [{{0,1,1,0},{1,0,1,1},{1,1,0,1},{0,1,1,0}},
VertexLabels →{1→a,2→b,3→c,4→d}]
```

Out[52]=



In[51]:=
```
graph5 =AdjacencyGraph [{{0,1,1,0},{1,0,1,1},{1,1,0,1},{0,1,1,0}},
VertexLabels →{1→a,2→b,3→c,4→d},DirectedEdges →True]
```

Out[51]=



Извеждане на списък на съседство

In[17]:=
```
AdjacencyList [graph5 ,#]&/@VertexList [graph5 ]
```

Out[17]= {{2, 3}, {1, 3, 4}, {1, 2, 4}, {2, 3}}

Извеждане на най-краткия път

In[18]:=
```
FindShortestPath [graph1 ,g,b]
```

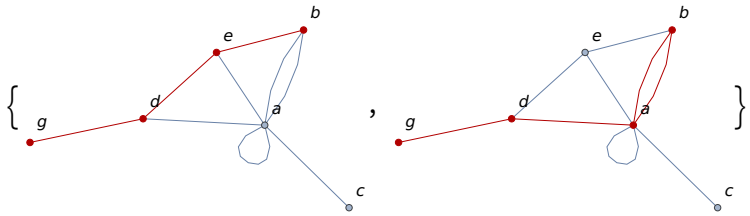
Out[18]= {g, d, a, b}

Извеждане на всички пътища с дължината на най-краткия път

In[19]:=
```
FindPath [graph1 ,g,b,{GraphDistance [graph1 ,g,b]},All]
HighlightGraph [graph1 ,PathGraph [#]]&/@%
```

Out[19]= {{g, d, e, b}, {g, d, a, b}}

Out[20]=



Извеждане на матрицата от всички разстояния

In[21]:=
```
GraphDistanceMatrix [graph1 ]//MatrixForm
```

Out[21]//MatrixForm=

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 2 \\ 1 & 0 & 2 & 2 & 1 & 3 \\ 1 & 2 & 0 & 2 & 2 & 3 \\ 1 & 2 & 2 & 0 & 1 & 1 \\ 1 & 1 & 2 & 1 & 0 & 2 \\ 2 & 3 & 3 & 1 & 2 & 0 \end{pmatrix}$$

Откриване на цикъл в граф

In[22]:=
```
FindCycle [graph1 ]
```

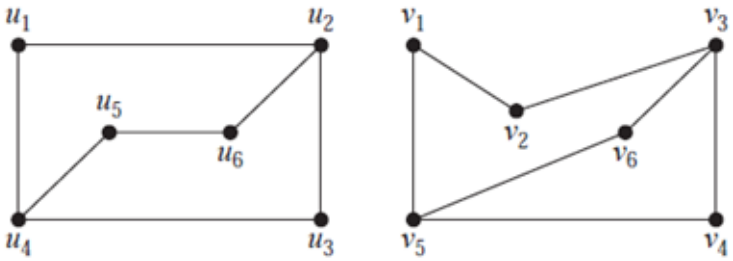
Out[22]= {{a ⟷ b, b ⟷ a}}

In[23]:=
```
FindCycle [graph5 ]
```

Out[23]= {{1 ⟷ 2, 2 ⟷ 1}}

Изоморфизъм

Задача: Изчертайте следните графи и проверете дали са изоморфни

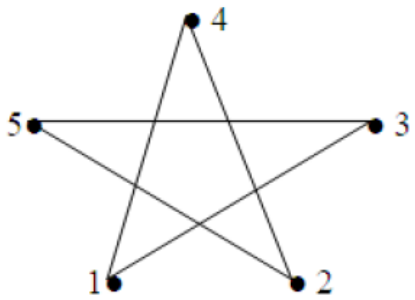



```
(*Esc ue Esc*)
```

In[53]:=
```
graph6 =Graph [{u1 ,u2 ,u3 ,u4 ,u5 ,u6},{u1⟷u2 ,u1⟷u4 ,u2⟷u3 ,u2⟷u6 ,u6⟷u5 ,u5⟷u4 ,u4⟷u3},
VertexLabels →"Name "];
graph7 =Graph [{v1 ,v2 ,v3 ,v4 ,v5 ,v6},{v1⟷v2 ,v1⟷v5 ,v2⟷v3 ,v3⟷v6 ,v6⟷v5 ,v5⟷v4 ,v3⟷v4},
VertexLabels →"Name "];
IsomorphicGraphQ [graph6 ,graph7 ]
```

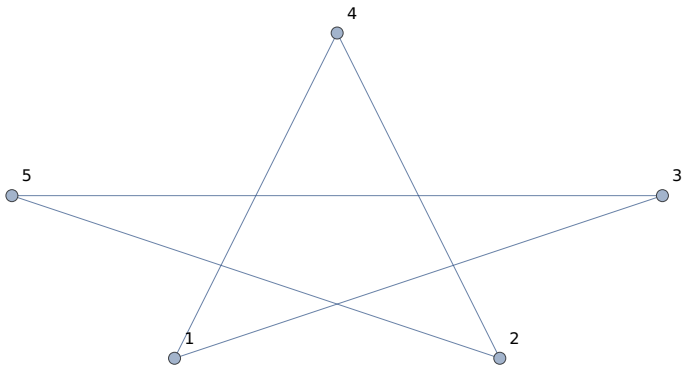
Out[55]= True

Ойлеров и Хамилтънов граф



In[56]:=
```
graph8 =Graph [{1 ,2 ,3 ,4 ,5},{1⟷3 ,1⟷4 ,2⟷5 ,2⟷4 ,3⟷5},VertexLabels →"Name "];
(*Задаване на координати за поставяне на всеки връх*)
graph81 =Graph [{1 ,2 ,3 ,4 ,5},{1⟷3 ,1⟷4 ,2⟷5 ,2⟷4 ,3⟷5},
VertexCoordinates → {1 → {2, 1}, 2 → {4, 1},3→{5,2},4→{3,3},5→{1,2}},
VertexLabels →"Name "]
EulerianGraphQ [graph8 ]
HamiltonianGraphQ [graph8 ]
```

Out[57]=



Out[58]= True

Out[59]= True

In[38]:=
```
FindEulerianCycle [graph8 ]
```

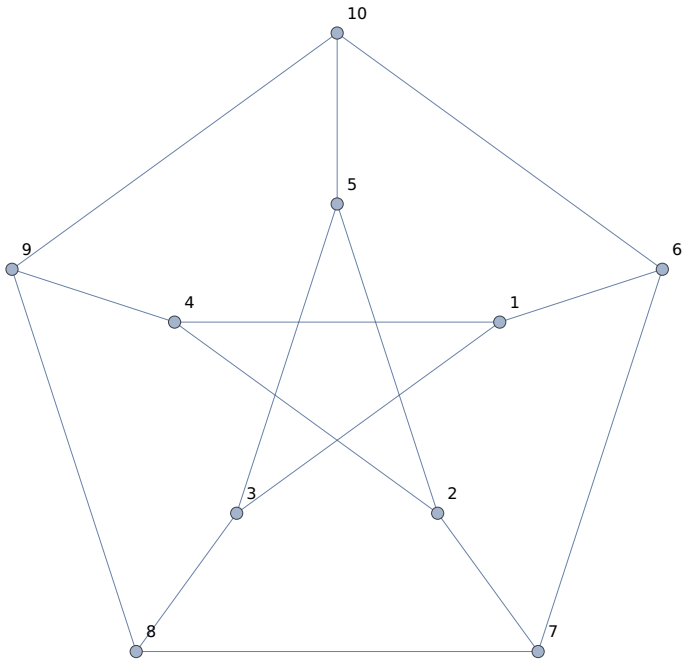
Out[38]= {{1 ⟷ 4, 4 ⟷ 2, 2 ⟷ 5, 5 ⟷ 3, 3 ⟷ 1}}

In[39]:=
```
FindHamiltonianCycle [graph8]
```

Out[39]= {{1 ⟷ 3, 3 ⟷ 5, 5 ⟷ 2, 2 ⟷ 4, 4 ⟷ 1}}

In[40]:=
```
grPeter =PetersenGraph [5,2,VertexLabels →"Name"]
```

Out[40]=



In[41]:=
```
EulerianGraphQ [grPeter]
```

Out[41]= False

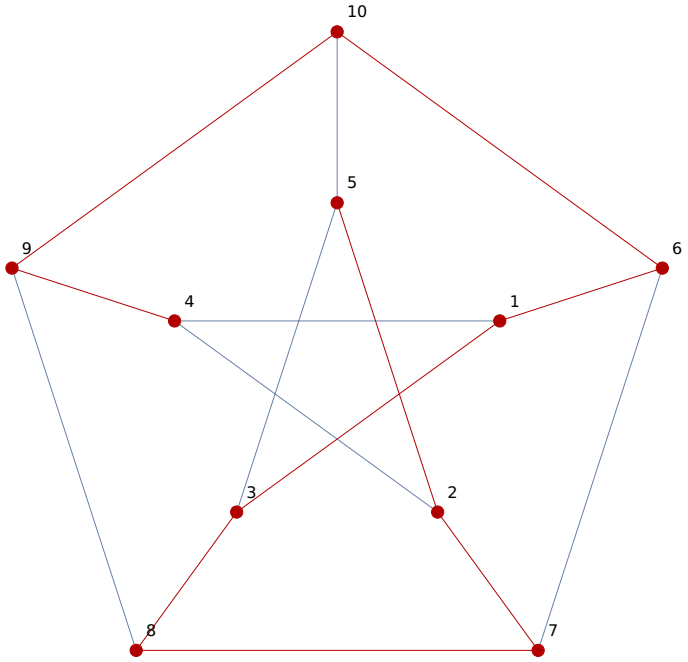
In[42]:=
```
HamiltonianGraphQ [grPeter]
```

Out[42]= False

In[43]:=
```
FindHamiltonianPath [grPeter ]
HighlightGraph [grPeter ,PathGraph [%]]
```

Out[43]= {4, 9, 10, 6, 1, 3, 8, 7, 2, 5}

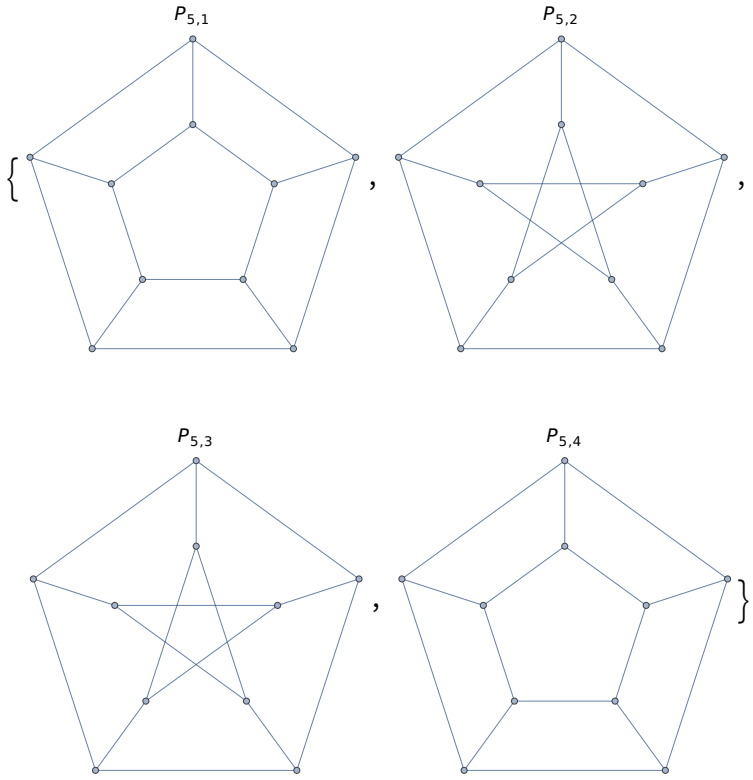
Out[44]=



Някои специални видове графи

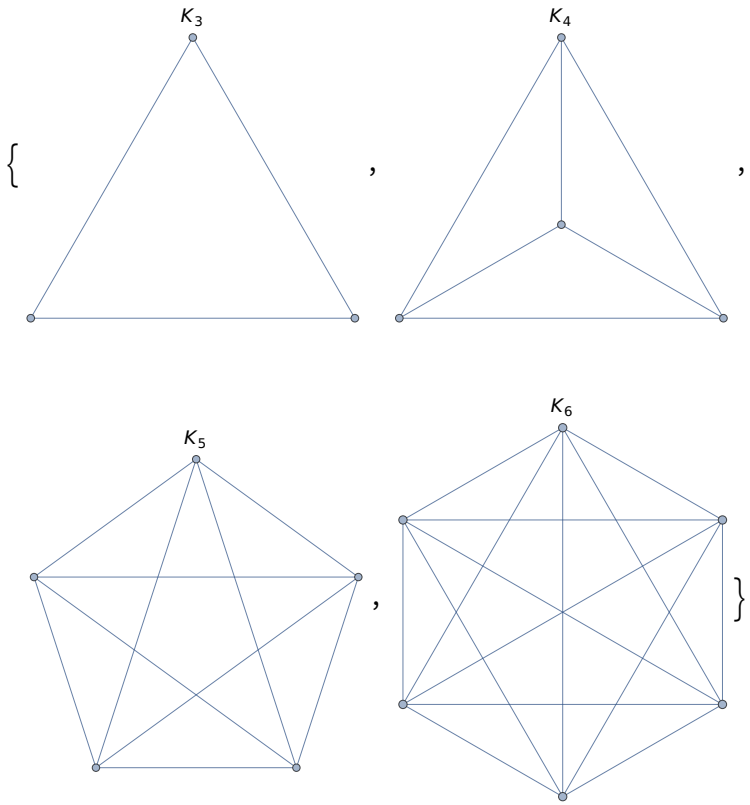
In[45]:=
```
Table[PetersenGraph [5, i, PlotLabel → Subscript [P, 5, i]], {i, 4}]
```

Out[45]=



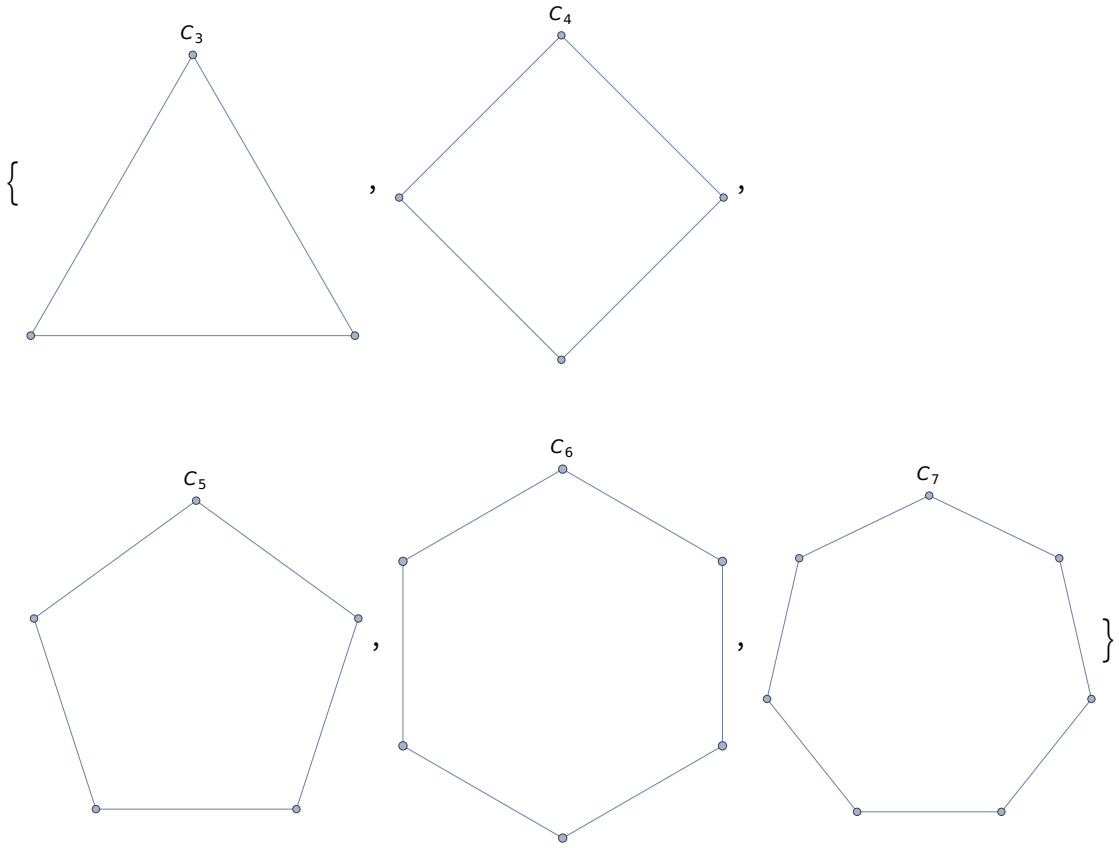
In[46]:=
```
Table[CompleteGraph [i, PlotLabel → Subscript [K, i]], {i, 3, 6}]
```

Out[46]=



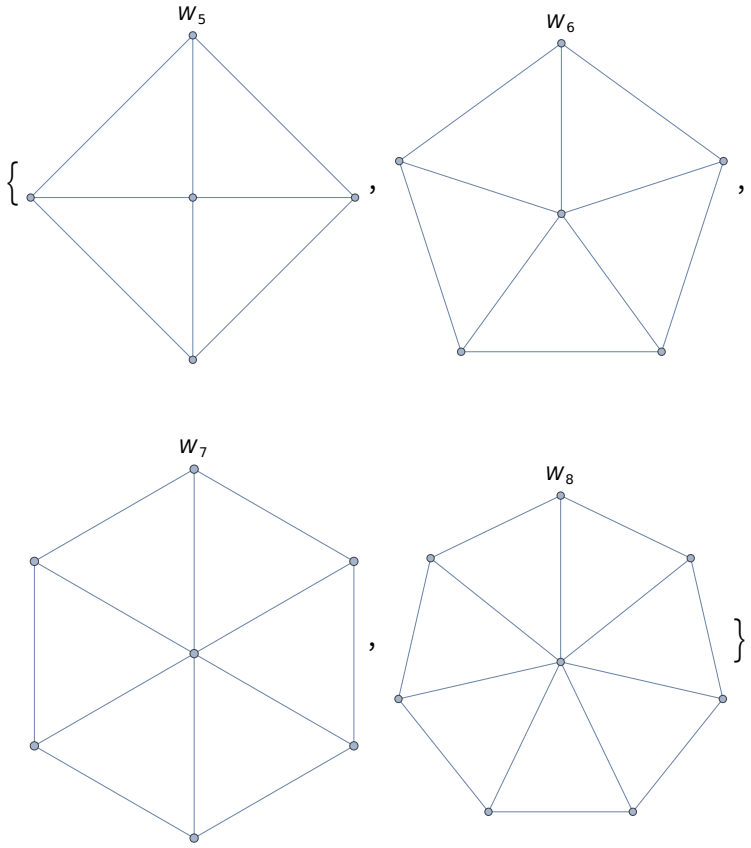
In[47]:=
```
Table[CycleGraph[i, PlotLabel → Subscript[C, i]], {i, 3,7}]
```

Out[47]=



In[48]:=
```
Table[WheelGraph[i, PlotLabel → Subscript[W, i]], {i, 5, 8}]
```

Out[48]=



In[49]:=
```
Table[HypercubeGraph[i, PlotLabel → Subscript[Q, i]], {i, 2, 4}]
```

Out[49]=