In[414]:=
```
inFileName = StringJoin[{NotebookDirectory[], "input.txt"}];
fileStream = OpenRead[inFileName];
vertex = Read[fileStream, {Word, Number}][[2]];
edge = Read[fileStream, {Word, Number}][[2]];
edges = ReadList[fileStream, Expression, edge];
vertexList = Array[# &, vertex];
edgesList = Table[edges[[i, 1]] ↔ edges[[i, 2]], {i, edge}];
listInput = ReadList[fileStream, String];
array = Array[0 &, vertex];
For[i = 1, i ≤ vertex, i++, array[[i]] =
    ToExpression[StringSplit[listInput[[i]], {"b", "_", "/*", "*/"}]][[2]]];
Close[fileStream];
graph = Graph[vertexList, edgesList, { GraphLayout → "CircularEmbedding",
     VertexSize → 0.3, VertexLabels → Placed["Name", Center],
     VertexLabelStyle → Directive[Bold, Italic, 20],
     EdgeShapeFunction → GraphElementData["Arrow", "ArrowSize" → 0.05]}];
equations = Array[0 &, vertex];
vars = Array[0 &, edge];
vars = Subscript[x, #] & /@ edgesList;
equations =
  Total /@ (Subscript[x, #] & /@ (edgesList // Cases[# ↔ _]) &) /@ vertexList -
    Total /@ (Subscript[x, #] & /@ (edgesList // Cases[_ ↔ #]) &) /@ vertexList;
Solve[equations == array, vars];
equations == array /. % // Simplify
```

упростить

```
equations = equations[[#]] == array[[#]] & /@ Range[vertex];
                                              диапазон
Row[{MatrixForm@equations, MatrixForm@array}, "="]
ряд  матричная форма        матричная форма
```

```
pred = ConstantArray[0, vertex];
       постоянный массив
depth = ConstantArray[0, vertex];
        постоянный массив
dir = ConstantArray[0, vertex];
      постоянный массив
listUt = {};
dinastVertex = {};
prev = root;
BuildSpanningTreeForGraph[g_, root_] := Module[{s = {}},
                                         программный модуль
    DepthFirstScan[UndirectedGraph[g], root,
    проход в глубину  ненаправленный граф
     {"FrontierEdge" -> Function[e, {AppendTo[s, e[[1]] ↦ e[[2]]],
                        функция      добавить в конец к
         pred[[e[[2]]]] = e[[1]], depth[[e[[2]]]] = 1 + depth[[e[[1]]]]}],
      "PrevisitVertex" → Function[u, AppendTo[dinastVertex, u]]}];
                         функция      добавить в конец к
    For[k = 1, k ≤ Length[s], k++, arc = s[[k]];
    цикл ДЛЯ          длина
     If[MemberQ[edgesList, arc], {dir[[arc[[2]]]] = 1;
     … элемент списка?
        AppendTo[listUt, arc]}, {dir[[arc[[2]]]] = -1;
        добавить в конец к
        AppendTo[listUt, Reverse[arc]]}];];
        добавить в конец к  расположить в обратном порядке
    Return[s];];
    вернуть управление
```

```
root = 5;
(*1,3*)
DFS = BuildSpanningTreeForGraph[graph, root];
(*2*)
Print["Множество дуг покрывающего дерева"]
печатать
listUt
Print["Множество дуг, которые не вошли в покрывающее дерево"]
печатать
listUn = Complement[edgesList, listUt] (*Un=U\Ut*)
         дополнение
(*4*)
```

```
graph
HighlightGraph[graph, listUt, VertexLabels → "Some"]
граф с подкраской                    метки для вершин
HighlightGraph[graph, listUn, VertexLabels → "Some"]
граф с подкраской                    метки для вершин
Print["Покрывающее дерево"]
печатать
Graph[listUt, GraphLayout → {"LayeredDigraphEmbedding", "RootVertex" → root},
граф            укладка графа
 VertexSize → 0.5, VertexLabels → Placed["Name", Center],
 размер вершины    метки для вершин расположен      центр
 VertexLabelStyle → Directive[Bold, Italic, 20],
 стиль меток вершин    директива   жи…  курсив
 EdgeShapeFunction → GraphElementData["Arrow", "ArrowSize" → 0.12]]
 функция формы ребра                  стрелка
Print["Корневое дерево с подсвеченным корнем"]
печатать
TreeGraph[DFS, GraphLayout → {"LayeredDigraphEmbedding", "RootVertex" → root},
граф дерево      укладка графа
 VertexSize → 0.5, VertexLabels → Placed["Name", Center],
 размер вершины    метки для вершин расположен      центр
 VertexStyle → {root → Red}, VertexLabelStyle → Directive[Bold, Italic, 20],
 стиль вершины          крас…  стиль меток вершин    директива   жи…  курсив
 EdgeShapeFunction → GraphElementData["Arrow", "ArrowSize" → 0.12]]
 функция формы ребра                  стрелка
all = {Prepend[vertexList, "vertex"], Prepend[pred, "pred"],
       добавить в начало               добавить в начало
   Prepend[dir, "dir"], Prepend[depth, "depth"], Prepend[dinastVertex, "d"]};
   добавить в начало    добавить в начало       добавить в начало
Print["Root = ", root]
печатать  корень уравнения
Text[Grid[all, Alignment -> Left, Spacings -> {2, 1}, Frame -> All]]
текст  таблица   выравнивание  слева  размер зазора      рамка   всё


treeEdges = {};
For[i = 1, i ≤ edge, i++,
цикл ДЛЯ
  For[j = 1, j ≤ Length[DFS], j++,
  цикл ДЛЯ          длина
   If[UndirectedEdge[edgesList[[i, 1]], edgesList[[i, 2]]] ==
   …  ненаправленное ребро
      UndirectedEdge[DFS[[j, 1]], DFS[[j, 2]]],
      ненаправленное ребро
     AppendTo[treeEdges, edgesList[[i]]]]
     добавить в конец к
    If[UndirectedEdge[edgesList[[i, 2]], edgesList[[i, 1]]] ==
    …  ненаправленное ребро
       UndirectedEdge[DFS[[j, 1]], DFS[[j, 2]]],
       ненаправленное ребро
      AppendTo[treeEdges, edgesList[[i]]]]];
      добавить в конец к
```

```
EquationsBalance[gArray_, gPred_, gDinastVertex_,
  gDir_, gEdgesList_, gTreeEdges_] := Module[{},
                                      программный модуль
  xp = ConstantArray[0, vertex];
       постоянный массив
  Map[{i = gDinastVertex[[#]],
  преобразовать
     xp[[i]] += -gDir[[i]] * gArray[[i]],
     xp[[gPred[[i]]]] += gDir[[i]] * gDir[[gPred[[i]]]] * xp[[i]]} &,
    Reverse[vertexList]];
    расположить в обратном порядке
  arr = Subscript[x, #1] → 0 & /@ Complement[edgesList, treeEdges];
        с нижним индексом          дополнение
  xij = List[];
        список
  For[i = 1, i ≤ vertex, i++,
  цикл ДЛЯ
    If[gDir[[i]] == 0, Continue[]] ×
    условный оператор   продолжить
     If [gDir[[i]] == 1, variable = gPred[[i]] ↦ i];
     условный оператор
    If[gDir[[i]] == -1, variable = i ↦ gPred[[i]]];
    условный оператор
    AppendTo[xij, Subscript[x, variable] → xp[[i]]];
    добавить в коне··  с нижним индексом
  ];
  result = Join[arr, xij];
           соединить
  Return[result];]
  вернуть управление

result = EquationsBalance[array, pred, dinastVertex, dir, edgesList, treeEdges]

Simplify[equations /. result]
упростить
fileName = FileNameJoin[{NotebookDirectory[], "5_lab.pdf"}];
           соединить пути   директория файла блокнота
NotebookFind[EvaluationNotebook[], "Output", All, CellStyle];
найти в блокн···  блокнот, содержащий выполняемое вы···  всё  стиль ячейки



rules = {};
τFunction[τ_, pos_] := (
If[dir[[τ]] == 1,
условный оператор
    AppendTo[rules, Flatten[{pos, Position[edgesList, pred[[τ]] ↦ τ]}] -> 1],
    добавить в конец к  уплостить  позиция по образцу
    AppendTo[rules, Flatten[{pos, Position[edgesList, τ ↦ pred[[τ]]]}] -> -1]]);
    добавить в конец к  уплостить  позиция по образцу
```

```
ρFunction[ρ_, pos_] := (
If[dir[[ρ]] == 1,
  AppendTo[rules, Flatten[{pos, Position[edgesList, pred[[ρ]] ↦ ρ]}] -> -1],
  AppendTo[rules, Flatten[{pos, Position[edgesList, ρ ↦ pred[[ρ]]]}] -> 1]]);
characteristic[edge_] := (
τ = Part[edge, 1]; ρ = Part[edge, 2];
position = Position[listUn, edge];
AppendTo[rules, Flatten[{position, Position[edgesList, edge]}] -> 1];
If[depth[[τ]] > depth[[ρ]],
  While[depth[[τ]] != depth[[ρ]], τFunction[τ, position];
   τ = pred[[τ]]], If[depth[[τ]] < depth[[ρ]],
  While[depth[[τ]] != depth[[ρ]], ρFunction[ρ, position];
   ρ = pred[[ρ]]]];
If[depth[[τ]] == depth[[ρ]], While[τ != ρ, τFunction[τ, position];
  τ = pred[[τ]];
  ρFunction[ρ, position]; ρ = pred[[ρ]]]];
Return[Graph[vertexList, Append[listUt, edge],
  VertexSize → 0.5, VertexLabels → Placed["Name", Center],
  VertexLabelStyle → Directive[Bold, Italic, 20],
  EdgeShapeFunction → GraphElementData["Arrow", "ArrowSize" → 0.05],
  GraphLayout -> "CircularEmbedding", GraphHighlight -> edge]];
);

Print["Spanning tree of the graph with cycles:"]
characteristic /@ listUn
δ = SparseArray[rules, {Length[listUn], Length[edgesList]}];
Print["Characteristic vectors:"]
TableForm[Normal[δ], TableHeadings -> {listUn, Subscript[x, #] & /@ edgesList}]
```

Solve: Equations may not give solutions for all "solve" variables.

Out[431]= {True}

Out[433]=
$$\begin{pmatrix} x_{1\bullet\to 3} + x_{1\bullet\to 7} - x_{2\bullet\to 1} - x_{5\bullet\to 1} == 7 \\ x_{2\bullet\to 1} + x_{2\bullet\to 5} == 4 \\ -x_{1\bullet\to 3} + x_{3\bullet\to 4} + x_{3\bullet\to 6} - x_{5\bullet\to 3} - x_{7\bullet\to 3} == -1 \\ -x_{3\bullet\to 4} + x_{4\bullet\to 7} - x_{5\bullet\to 4} == -7 \\ -x_{2\bullet\to 5} + x_{5\bullet\to 1} + x_{5\bullet\to 3} + x_{5\bullet\to 4} - x_{6\bullet\to 5} - x_{7\bullet\to 5} == -2 \\ -x_{3\bullet\to 6} + x_{6\bullet\to 5} == -2 \\ -x_{1\bullet\to 7} - x_{4\bullet\to 7} + x_{7\bullet\to 3} + x_{7\bullet\to 5} == 1 \end{pmatrix} = \begin{pmatrix} 7 \\ 4 \\ -1 \\ -7 \\ -2 \\ -2 \\ 1 \end{pmatrix}$$

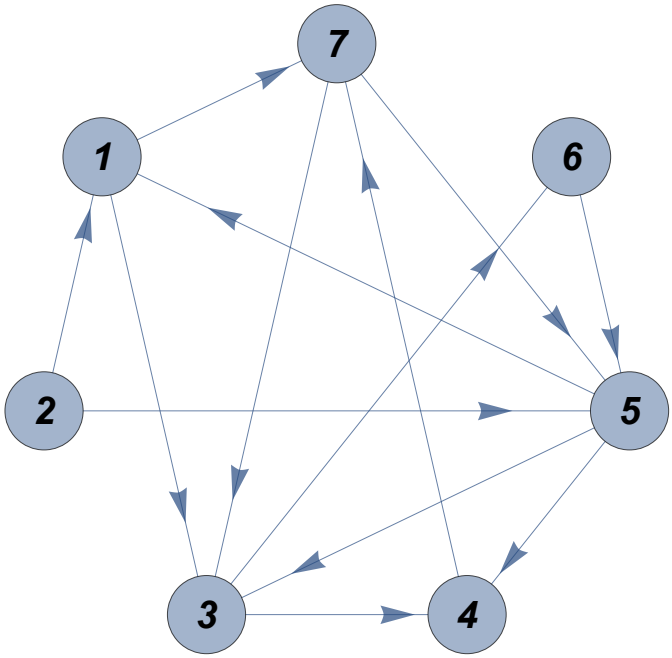
Множество дуг покрывающего дерева

Out[444]= $\{2 \bullet\to 5,\ 2 \bullet\to 1,\ 1 \bullet\to 3,\ 7 \bullet\to 3,\ 4 \bullet\to 7,\ 3 \bullet\to 6\}$

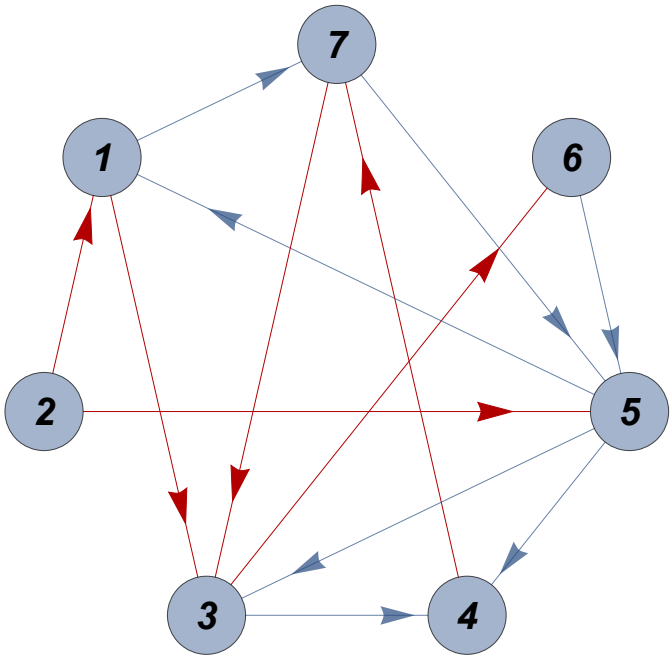
Множество дуг, которые не вошли в покрывающее дерево

Out[446]= $\{1 \bullet\to 7,\ 3 \bullet\to 4,\ 5 \bullet\to 1,\ 5 \bullet\to 3,\ 5 \bullet\to 4,\ 6 \bullet\to 5,\ 7 \bullet\to 5\}$

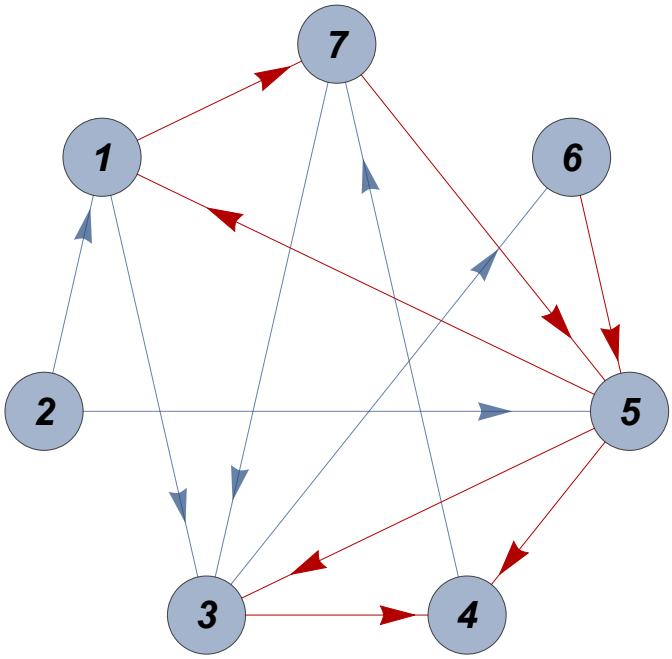
Out[447]=



Out[448]=

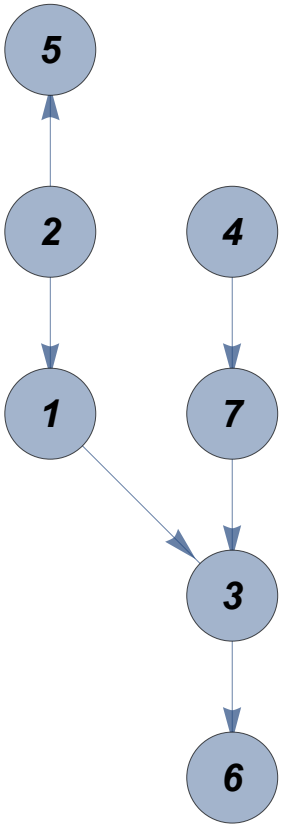


Out[449]=



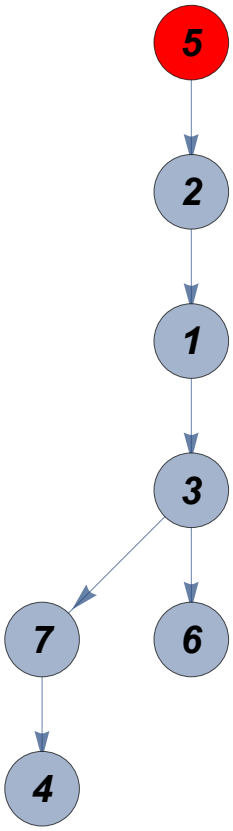
Покрывающее дерево

Out[451]=



Корневое дерево с подсвеченным корнем

Out[453]=



Root = 5

Out[456]=

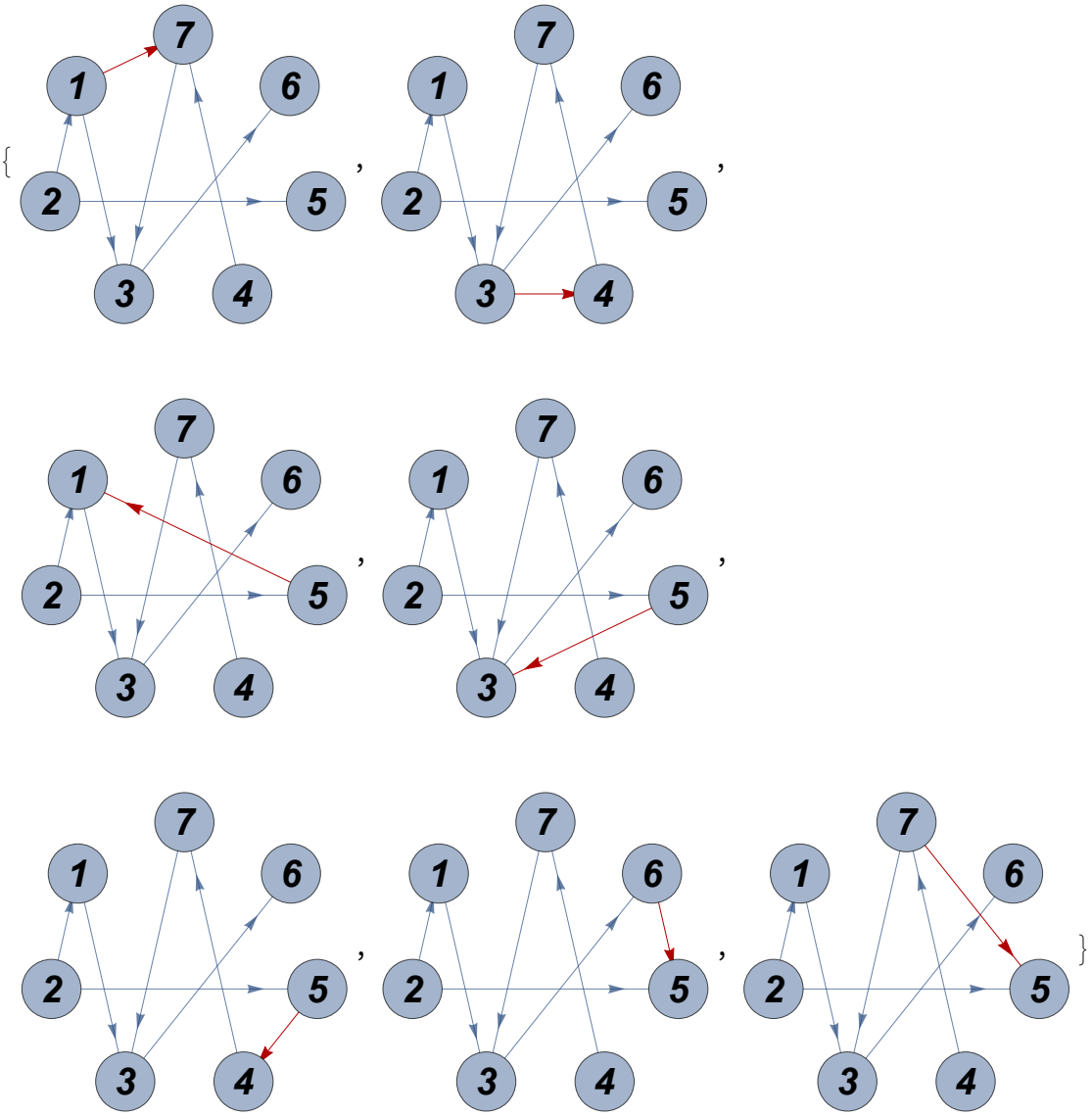
vertex	1	2	3	4	5	6	7
pred	2	5	1	7	0	3	3
dir	1	−1	1	−1	0	1	−1
depth	2	1	3	5	0	4	4
d	5	2	1	3	7	4	6

Out[460]= $\{x_{1\leftrightarrow 7} \to 0,\ x_{3\leftrightarrow 4} \to 0,\ x_{5\leftrightarrow 1} \to 0,\ x_{5\leftrightarrow 3} \to 0,\ x_{5\leftrightarrow 4} \to 0,\ x_{6\leftrightarrow 5} \to 0,$
$x_{7\leftrightarrow 5} \to 0,\ x_{2\leftrightarrow 1} \to 2,\ x_{2\leftrightarrow 5} \to 2,\ x_{1\leftrightarrow 3} \to 9,\ x_{4\leftrightarrow 7} \to -7,\ x_{3\leftrightarrow 6} \to 2,\ x_{7\leftrightarrow 3} \to -6\}$

Out[461]= {True, True, True, True, True, True, True}

Spanning tree of the graph with cycles:

Out[469]=



Characteristic vectors:

Out[472]//TableForm=

	$x_{1\leftrightarrow 7}$	$x_{1\leftrightarrow 3}$	$x_{2\leftrightarrow 1}$	$x_{2\leftrightarrow 5}$	$x_{3\leftrightarrow 6}$	$x_{3\leftrightarrow 4}$	$x_{4\leftrightarrow 7}$	$x_{5\leftrightarrow 1}$	$x_{5\leftrightarrow 3}$	$x_{5\leftrightarrow 4}$
$1 \leftrightarrow 7$	1	−1	0	0	0	0	0	0	0	0
$3 \leftrightarrow 4$	0	0	0	0	0	1	1	0	0	0
$5 \leftrightarrow 1$	0	0	−1	1	0	0	0	1	0	0
$5 \leftrightarrow 3$	0	−1	−1	1	0	0	0	0	1	0
$5 \leftrightarrow 4$	0	−1	−1	1	0	0	1	0	0	1
$6 \leftrightarrow 5$	0	1	1	−1	1	0	0	0	0	0
$7 \leftrightarrow 5$	0	1	1	−1	0	0	0	0	0	0