
Dunaev Viktor, 3 kurs, 6 group, 23 variant

Pre - Task

```
In[1]:= fname = NotebookDirectory[] <> "input.txt"
Out[1]= C:\6_Cemestr\DS_Laguto\input.txt

In[2]:= stream = OpenRead[fname];

In[3]:= information = ReadList[stream, String]
Out[3]= { /* | I | */ 6, /* | U | */ 12, {1,5}, {2,5}, {3,1}, {3,4},
        {3,5}, {4,1}, {4,2}, {4,6}, {5,4}, {6,2}, {6,3}, {6,5}, /*b_1*/ 7,
        /*b_2*/ 4, /*b_3*/ -1, /*b_4*/ -7, /*b_5*/ -2, /*b_6*/ -1}

In[4]:= numVertex = Read[StringToStream[StringSplit[information[[1]]][2]], Number]
Out[4]= 6

In[5]:= MyVertex = Table[i, {i, 1, numVertex}]
Out[5]= {1, 2, 3, 4, 5, 6}

In[6]:= numEdges = Read[StringToStream[StringSplit[information[[2]]][2]], Number]
Out[6]= 12

In[7]:= MyEdges = Table[
    list = StringSplit[information[[i]], {"{", "}", ","}];
    Read[StringToStream[list[[1]], Number] → Read[StringToStream[list[[2]], Number],
    {i, 3, 2 + numEdges}
]
Out[7]= {1 → 5, 2 → 5, 3 → 1, 3 → 4, 3 → 5, 4 → 1, 4 → 2, 4 → 6, 5 → 4, 6 → 2, 6 → 3, 6 → 5}

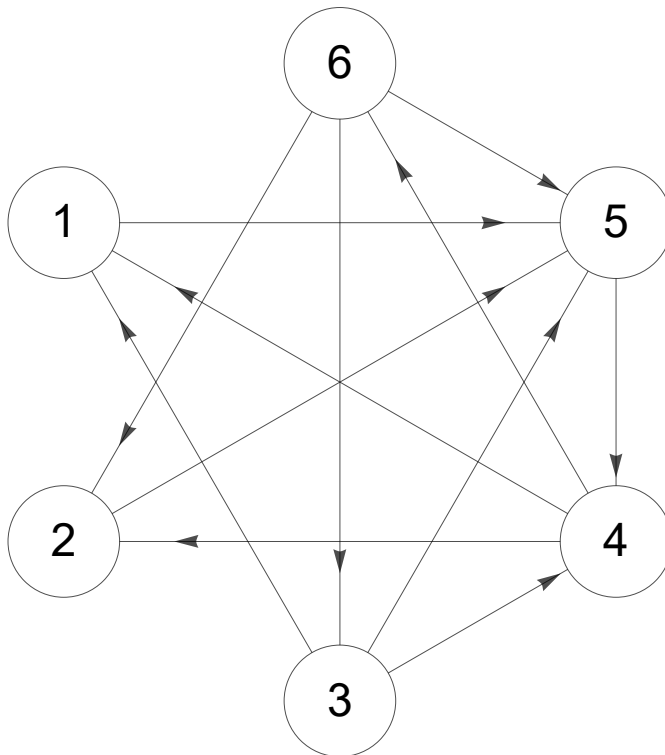
In[8]:= Close[stream]
Out[8]= C:\6_Cemestr\DS_Laguto\input.txt
```

```

In[9]:= myGraph = Graph[MyVertex, MyEdges, VertexLabels -> Placed["Name", Center],
  GraphLayout -> "CircularEmbedding", VertexSize -> 0.35, VertexStyle -> White,
  EdgeShapeFunction -> GraphElementData["Arrow", "ArrowSize" -> 0.035],
  EdgeStyle -> Black, VertexLabelStyle -> Large]

```

Out[9]=



Task I

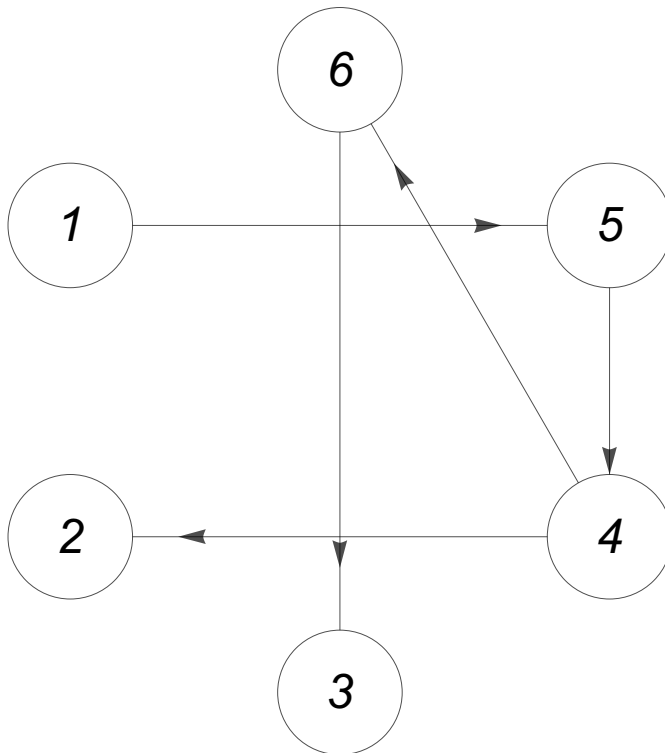
(* построить покрывающее дерево для индивидуального графа,отобразить его в задании *)

```

In[10]:= spanningTree = FindSpanningTree[myGraph, VertexSize -> Large,
  VertexStyle -> White, VertexLabels -> Placed["Name", Center],
  VertexLabelStyle -> Directive[Black, Italic, 25], GraphLayout ->
  {"CircularEmbedding"}, EdgeShapeFunction -> "Arrow", EdgeStyle -> Black]

```

Out[10]=



Task 2

(* выбрать произвольно узел-корень дерева, перейти к корневому дереву:
 построить списковые структуры
 хранения корневого дерева (список узлов, список предков,
 список направлений, список глубин узлов, список связи или династического обхода *)

```

In[11]:= startNode = RandomChoice[MyVertex]

```

Out[11]= 2

```

In[12]:= dirs = Table[0, Length[MyVertex]]

```

Out[12]= {0, 0, 0, 0, 0, 0}

```

In[13]:= pred = Table[0, Length[MyVertex]]

```

Out[13]= {0, 0, 0, 0, 0, 0}

```

In[14]:= depth = Table[0, Length[MyVertex]]

```

Out[14]= {0, 0, 0, 0, 0, 0}

```

In[15]:= dinast = {}

```

Out[15]= {}

```
In[16]:= tree = {}
```

```
Out[16]= {}
```

```
In[17]:= graphTree = {}
```

```
Out[17]= {}
```


```
In[18]:= DepthFirstScan[UndirectedGraph@myGraph, startNode,
  {"FrontierEdge" → ((edge = # /. (x_ ↔ y_) → (x → y); AppendTo[tree, edge];
    If[MemberQ[MyEdges, edge], dirs[[#2]] = 1;
    AppendTo[graphTree, edge], dirs[[#2]] = -1;
    AppendTo[graphTree, #2 ↔ #1]]];
  pred[[#2]] = #1];
  depth[[#2]] = depth[[#1]] + 1) &], "PrevisitVertex" → ((AppendTo[dinast, #] &))]
```

```
Out[18]= {5, 2, 4, 2, 6, 3}
```

```
In[19]:= tree
```

```
Out[19]= {2 ↔ 4, 4 ↔ 3, 3 ↔ 6, 6 ↔ 5, 5 ↔ 1}
```

```
In[29]:= Graph[tree, VertexSize -> Large,
  VertexStyle -> White, VertexLabels -> Placed["Name", Center],
  VertexLabelStyle -> Directive[Black, Italic, 25],
  GraphLayout -> {"RadialEmbedding"}, EdgeShapeFunction -> "Arrow", EdgeStyle -> Black]
```

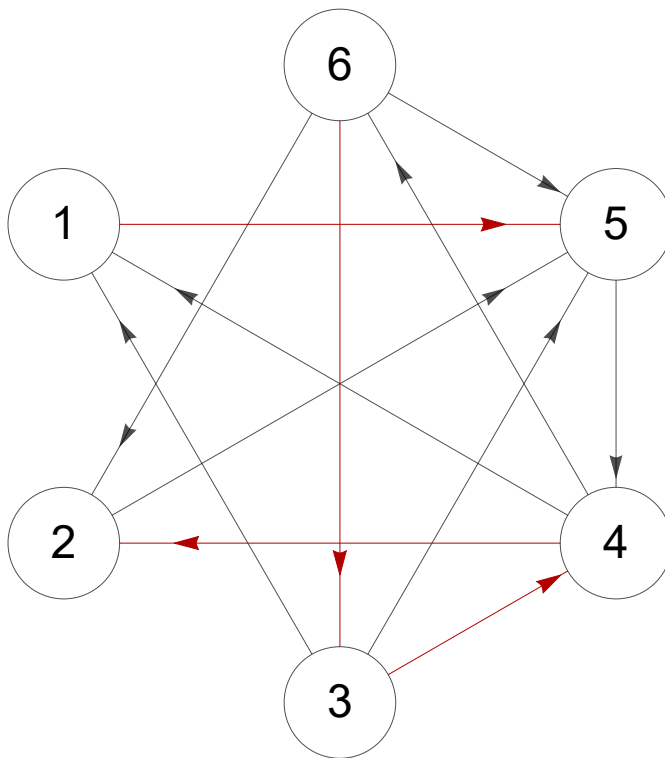
```
Out[29]= 
```

Task 3

(* вывести исходный граф с подсвеченными дугами покрывающего дерева *)

```
In[20]:= HighlightGraph[myGraph, graphTree]
```

```
Out[20]=
```



Task 4

(* списковые структуры вывести в виде таблицы (список связи или династического обхода можно не встраивать в таблицу, а вывести отдельным списком) *)

```
In[25]:= TableForm[{MyVertex, pred, depth, dirs},
  TableHeadings -> {"Вершины:", "Предки:", "Глубина:", "Направления:"}]
```

```
Out[25]//TableForm=
```

Вершины:	1	2	3	4	5	6
Предки:	5	0	4	2	6	3
Глубина:	5	0	2	1	4	3
Направления:	-1	0	-1	-1	-1	-1

```
In[26]:= dinast
```

```
Out[26]= {2, 4, 3, 6, 5, 1}
```