



МИЭТ

Национальный исследовательский университет «МИЭТ»

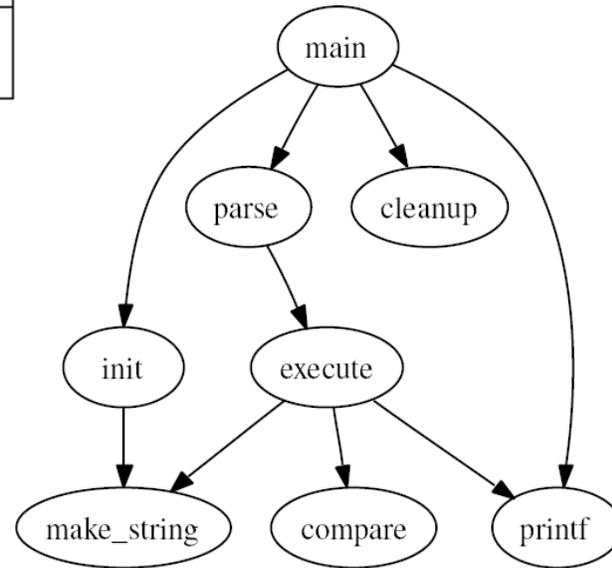
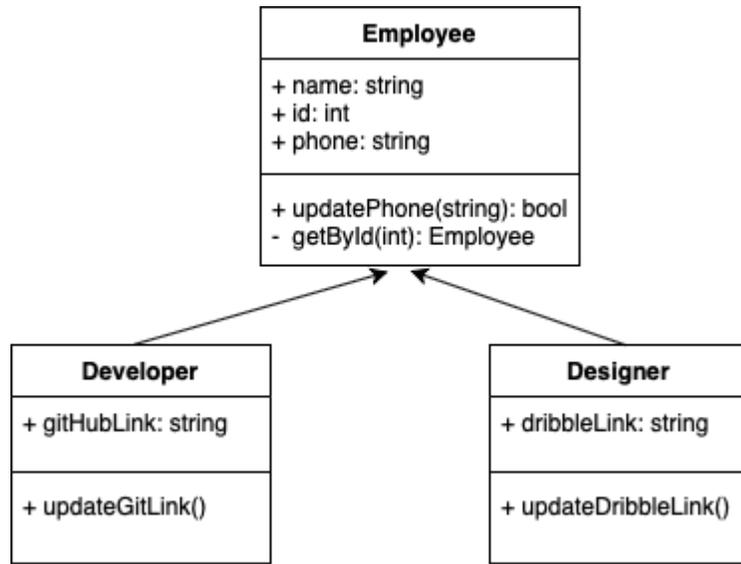
Кафедра ПКИМС

Компьютерные технологии в научных исследованиях

Семинар №5

Работа с пакетом graphviz

Задача визуализации графов



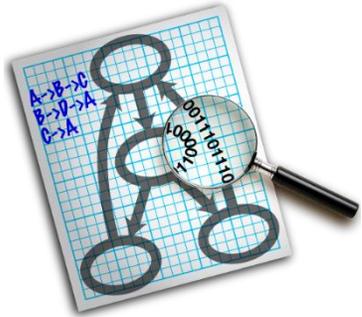
The screenshot shows the **INTRO V1.0** documentation interface. The **Files** tab is active, displaying the **Accel.c File Reference**. The page includes a navigation sidebar on the left with a search bar and a list of files. The main content area shows the source code for `Accel.c` with the following includes:

```
#include "Platform.h"
#include "Accel.h"
#include "UTIL1.h"
#include "ACCEL1.h"
#include "FSSH1.h"
```

Below the code, a dependency graph for `Accel.c` is shown, illustrating the relationships between `Accel.c`, `Accel.h`, `Platform.h`, `UTIL1.h`, `ACCEL1.h`, `FSSH1.h`, `PE_Types.h`, and `Cpu.h`. A tooltip for `Accel.h` states: "This is the interface to Accelerometer Module." Below the graph, the **Functions** section lists `void ACCEL_GetValues(int16_t *x, int16_t *y, int16_t *z)` with the description: "Returns the current accelerometer sensor values." The footer indicates the documentation was generated on Sat Jun 23 2012 23:18:14 for INTRO by **doxygen** 1.7.5.1.

Программа Graphviz

URL: <https://graphviz.org>



Download | Graphviz

graphviz.org/download/

Graphviz

Download Documentation Gallery Forum GitLab Search this site...

Search this site

Graphviz

About

Download

Source Code

Documentation

DOT Language

Command Line

Layout Engines

Output Formats

Attributes

Attribute Types

Graph

Attributes

Node Attributes

Node Shapes

Cluster

- Stable Windows install packages, built with Microsoft Visual Studio 16 2019:
 - graphviz-9.0.0
 - graphviz-9.0.0 (32-bit) ZIP archive [sha256] (contains all tools and libraries)
 - graphviz-9.0.0 (64-bit) EXE installer [sha256]
 - graphviz-9.0.0 (32-bit) EXE installer [sha256]
 - graphviz-8.1.0
 - graphviz-8.1.0 (32-bit) ZIP archive [sha256] (contains all tools and libraries)
 - graphviz-8.1.0 (64-bit) EXE installer [sha256]
 - graphviz-8.1.0 (32-bit) EXE installer [sha256]
 - graphviz-8.0.5
 - graphviz-8.0.5 (32-bit) ZIP archive [sha256] (contains all tools and libraries)
 - graphviz-8.0.5 (64-bit) EXE installer [sha256]
 - graphviz-8.0.5 (32-bit) EXE installer [sha256]
 - graphviz-8.0.3
 - graphviz-8.0.3 (32-bit) ZIP archive [sha256] (contains all tools and libraries)
 - graphviz-8.0.3 (64-bit) EXE installer [sha256]
 - graphviz-8.0.3 (32-bit) EXE installer [sha256]

View page source

Edit this page

Create child page

Create documentation issue

Print entire section

Source Code

Executable Packages

Linux

Windows

Mac

Solaris

Other Unix

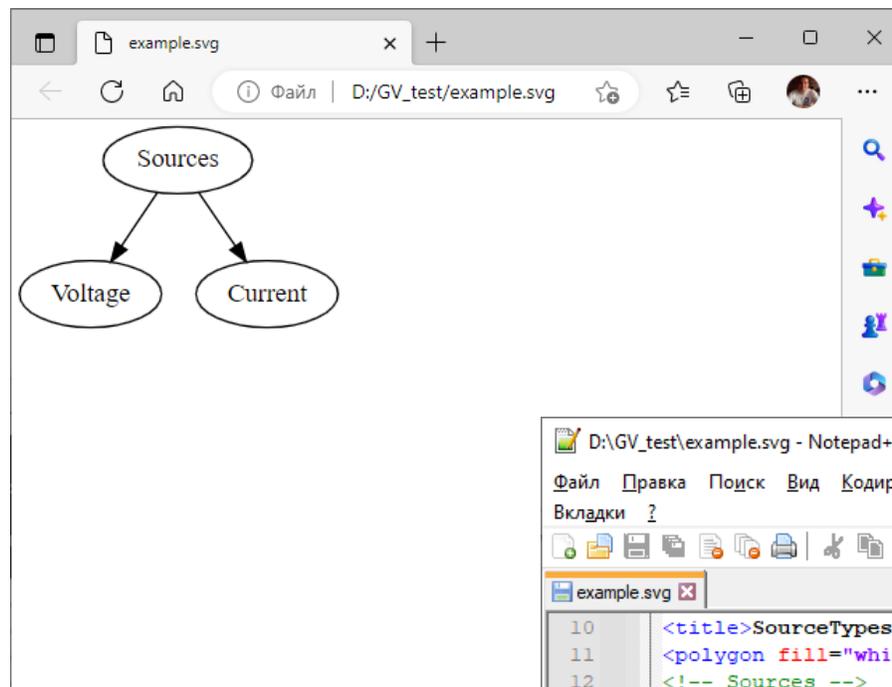
Выходные форматы GraphViz

Форматы изображений:

- bmp
- gif
- ico
- jpeg
- png
- psd
- tga
- tiff
- webp

Текстовые форматы изображений:

- fig
- pic
- ps
- svg



dot

-Tpng

<имя файла с графом>

-O

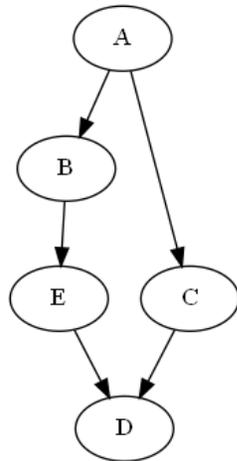
<имя файла с результатом>

```
D:\GV_test\example.svg - Notepad++
Файл  Правка  Поиск  Вид  Кодировки  Синтаксисы  Опции  Инструменты  Макросы  Запуск  Плагины
Вкладки  ?
example.svg x
10  <title>SourceTypes</title>
11  <polygon fill="white" stroke="transparent" points="-4,4 -4,-112 175.69,-1
12  <!-- Sources -->
13  <g id="node1" class="node">
14  <title>Sources</title>
15  <ellipse fill="none" stroke="black" cx="85.35" cy="-90" rx="40.09" ry="18
16  <text text-anchor="middle" x="85.35" y="-86.3" font-family="Times New Rom
17  </g>
18  <!-- Voltage -->
19  <g id="node2" class="node">
20  <title>Voltage</title>
21  <ellipse fill="none" stroke="black" cx="38.35" cy="-18" rx="38.19" ry="18
22  <text text-anchor="middle" x="38.35" y="-14.3" font-family="Times New Rom
23  </g>
24  <!-- Sources&#45;&gt;Voltage -->
25  <g id="edge1" class="edge">
26  <title>Sources&#45;&gt;Voltage</title>
```

Синтаксис формата dot

граф	::=	[strict] (graph digraph) [имя_графа] '{' список_объявлений '}'
список_объявлений	::=	[объявление ';' список_объявлений]
объявление	::=	подграф узел ребро атрибут имя '=' имя
подграф	::=	[subgraph [имя_подграфа]] '{' список_объявлений '}'
узел	::=	имя_узла [список_атрибутов]
ребро	::=	(имя_узла имя_подграфа) данные_ребра [список_атрибутов]
данные_ребра	::=	тип_связи (имя_узла имя_подграфа) [данные_ребра]
тип_связи	::=	(-- ->)

```
digraph {  
  A -> B;  
  A -> C -> D;  
  B -> E -> D;  
}
```

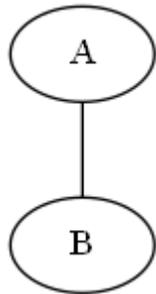


Типы графов: направленный и ненаправленный (1)

graph

Создаёт неориентированный граф:

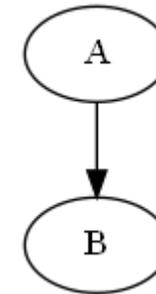
```
graph {  
    A -- B  
}
```



digraph

Создаёт ориентированный граф

```
digraph {  
    A -> B  
}
```



Описание графов

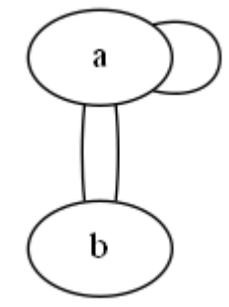
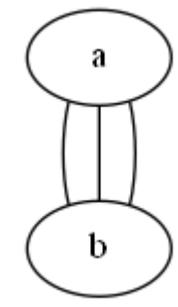
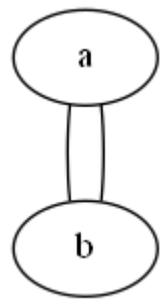
```
graph {  
  a -- b  
  a -- b  
}
```



```
graph {  
  a -- b  
  b -- a  
}
```

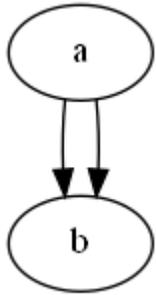
```
graph {  
  a -- b  
  a -- b  
  b -- a  
}
```

```
graph {  
  a -- b  
  a -- a  
  b -- a  
}
```

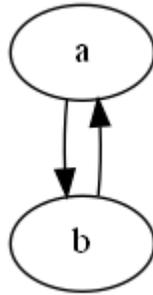


Описание орграфов

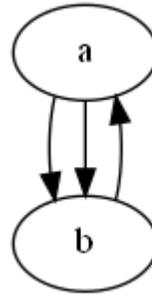
```
digraph {  
  a -> b  
  a -> b  
}
```



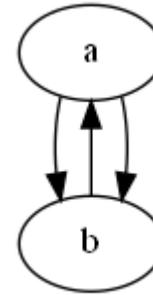
```
digraph {  
  a -> b  
  b -> a  
}
```



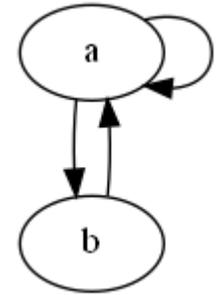
```
digraph {  
  a -> b  
  a -> b  
  b -> a  
}
```



```
digraph {  
  a -> b  
  b -> a  
  a -> b  
}
```



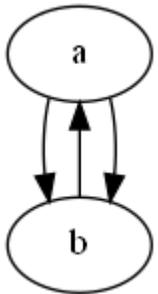
```
digraph {  
  a -> b  
  b -> a  
  a -> a  
}
```



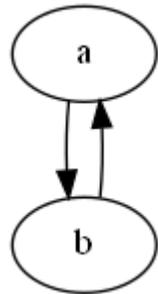
Ограничение числа рёбер: ключевое слово strict

Пример для ориентированных графов

```
digraph {  
  a -> b  
  b -> a  
  a -> b  
}
```

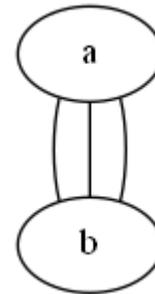


```
strict digraph {  
  a -> b  
  b -> a  
  a -> b  
}
```

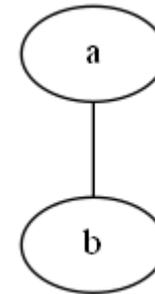


Пример для неориентированных графов

```
graph {  
  a -- b  
  a -- b  
  b -- a  
}
```

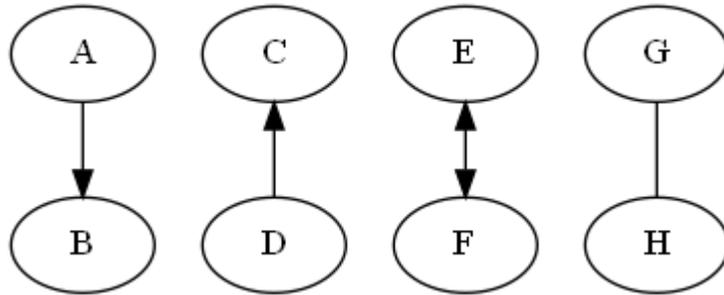


```
strict graph {  
  a -- b  
  a -- b  
  b -- a  
}
```

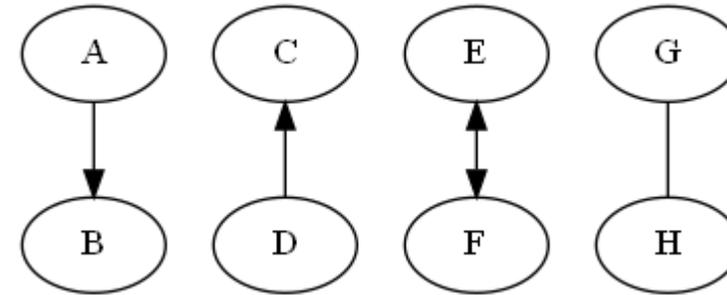


Атрибуты. Явное задание ориентации ребра (3)

```
digraph {  
  a -> b [dir=forward]  
  c -> d [dir=back]  
  e -> f [dir=both]  
  g -> h [dir=none]  
}
```



```
graph {  
  a -- b [dir=forward]  
  c -- d [dir=back]  
  e -- f [dir=both]  
  g -- h [dir=none]  
}
```



Возможные значения атрибута dir (применим только для рёбер):

forward - значение по умолчанию для орграфов

back

both

none - значение по умолчанию для неориентированных графов

Описание для группы рёбер

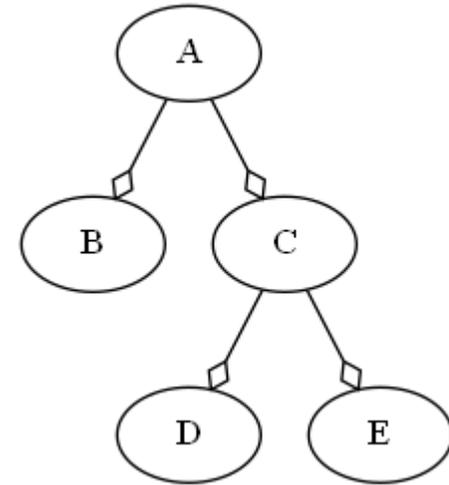
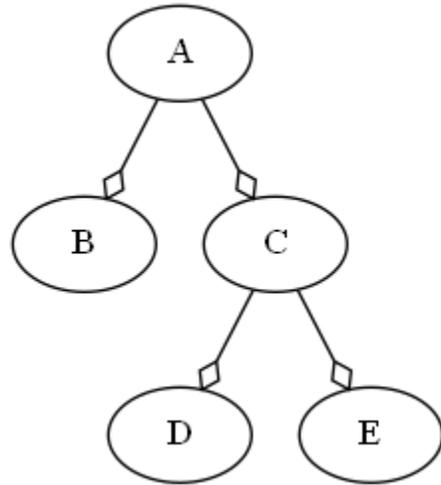
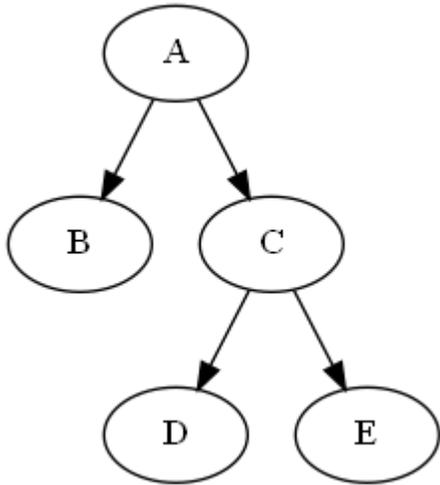
```
digraph {  
  A -> B  
  A -> C  
  C -> D  
  C -> E  
}
```



```
digraph {  
  A -> B [arrowhead=odiamond]  
  A -> C [arrowhead=odiamond]  
  C -> D [arrowhead=odiamond]  
  C -> E [arrowhead=odiamond]  
}
```

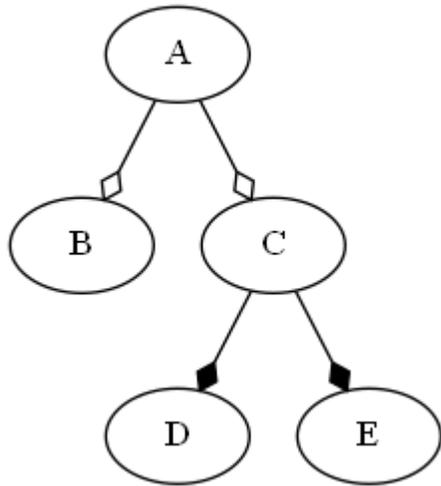


```
digraph {  
  edge [arrowhead=odiamond]  
  A -> B  
  A -> C  
  C -> D  
  C -> E  
}
```

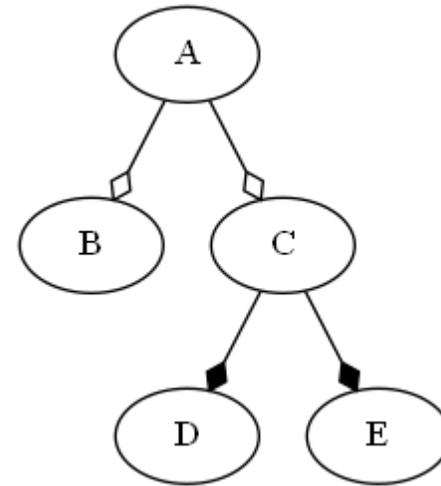


Описание рёбер (2)

```
digraph {  
  edge [arrowhead=odiamond]  
  A -> B  
  A -> C  
  C -> D [arrowhead=diamond]  
  C -> E [arrowhead=diamond]  
}
```

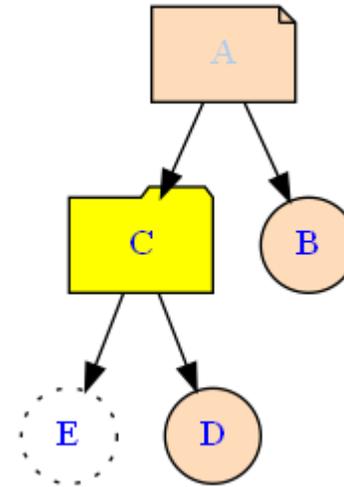


```
digraph {  
  edge [arrowhead=odiamond]  
  A -> B  
  A -> C  
  edge [arrowhead=diamond]  
  C -> D  
  C -> E  
}
```



Задание вида узлов (2)

```
digraph {  
  node [fontcolor=blue shape=circle style=filled fillcolor="#fedcba"]  
  
  A [fontcolor="#abcdef" shape=note]  
  C [shape=folder fillcolor="#ffff00"]  
  E [style=dotted]  
  
  A -> B  
  A -> C  
  C -> D  
  C -> E  
}
```



Изменение ориентации графа

```
digraph A {
```

```
rankdir=LR;
```

```
node [shape=egg]
```

```
A
```

```
node [shape=box]
```

```
B; C; D
```

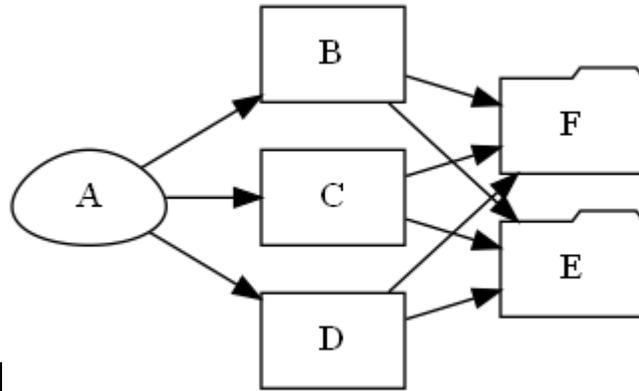
```
node [shape=folder]
```

```
F; E
```

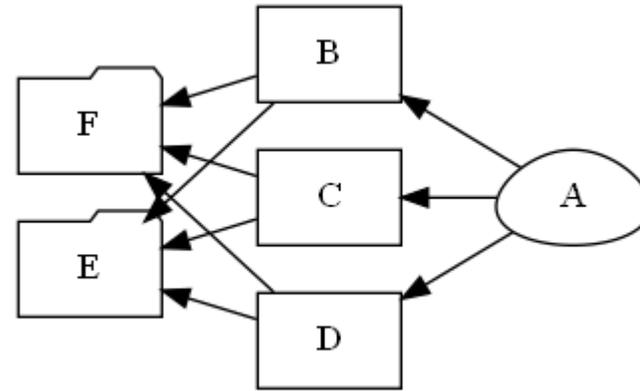
```
A -> {B, C, D} -> {F, E}
```

```
}
```

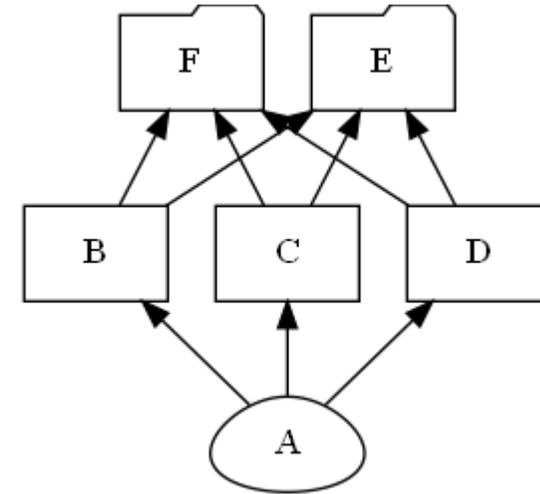
rankdir=LR;



rankdir=RL;



rankdir=BT;

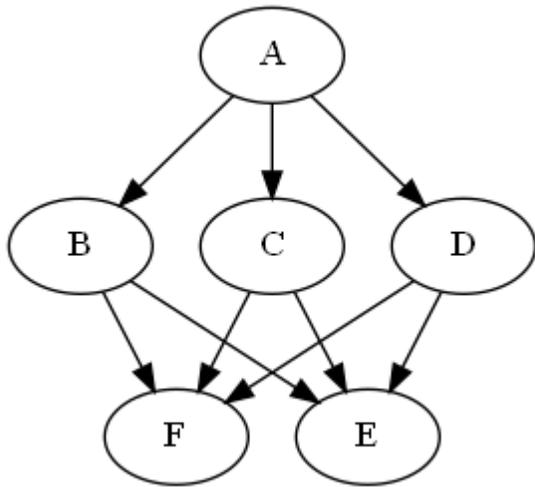


Группирование узлов (2)

```
digraph {
```

```
  A -> {B, C, D} -> F
```

```
}
```



```
digraph {
```

```
  node [shape=egg]
```

```
  A
```

```
  node [shape=box]
```

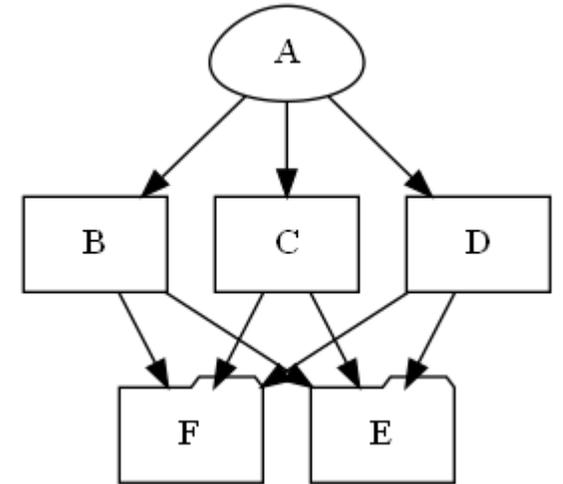
```
  B; C; D
```

```
  node [shape=folder]
```

```
  F; E
```

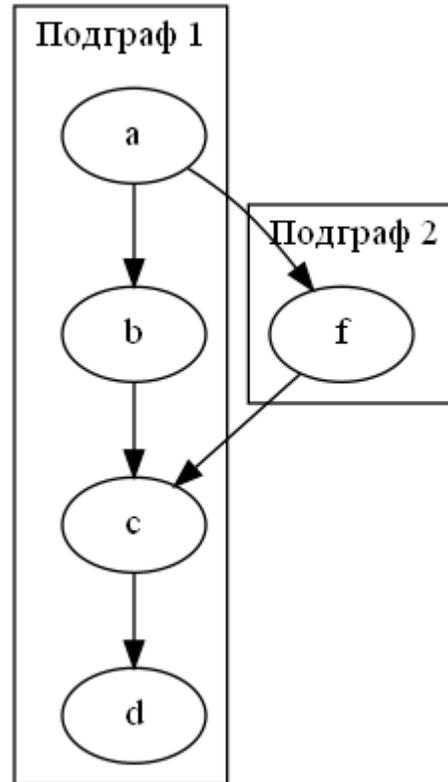
```
  A -> {B, C, D} -> {F, E}
```

```
}
```



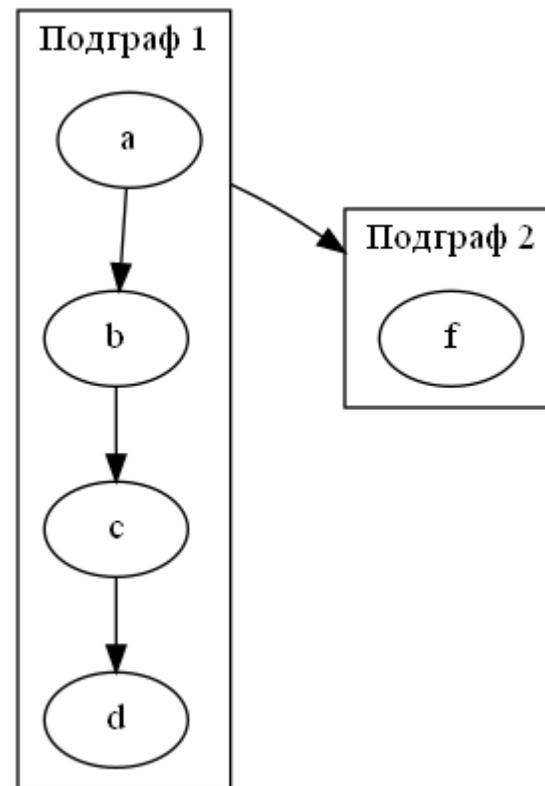
Группирование узлов подграфами (1)

```
digraph {  
  subgraph cluster_0 {  
    label="Подграф 1";  
    a -> b;  
    b -> c;  
    c -> d;  
  }  
  
  subgraph cluster_1 {  
    label="Подграф 2";  
    a -> f;  
    f -> c;  
  }  
}
```



Группирование узлов подграфами (2)

```
digraph {  
  graph [compound=true nodesep=1 ranksep=0.5]  
  subgraph cluster_0 {  
    label="Подграф 1";  
    a -> b;  
    b -> c;  
    c -> d;  
  }  
  
  subgraph cluster_1 {  
    label="Подграф 2";  
    f;  
  }  
  a -> f [ltail=cluster_0 lhead=cluster_1]  
}
```



`compound=true`

- разрешить связывать подграфы

`nodesep=1`

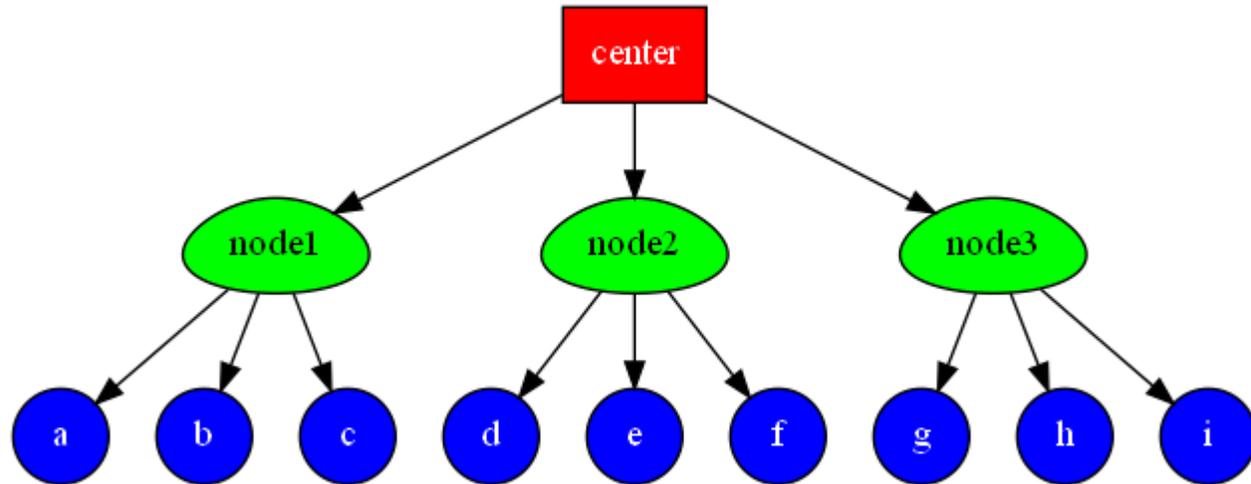
- длина ребра между подграфами

`ranksep=0.5`

- длина ребра внутри подграфа

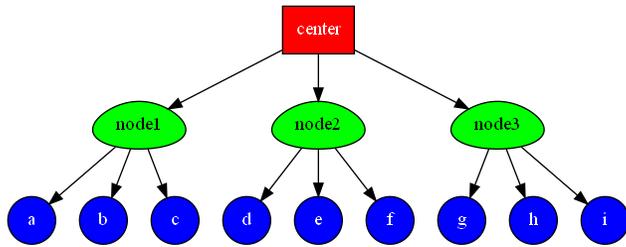
Алгоритмы визуализации графа (1)

```
digraph {  
  node [style=filled fillcolor=red shape=box fontcolor=white]  
  center  
  node [style=filled fillcolor=green shape=egg fontcolor=black]  
  node1; node2; node3  
  node [style=filled fillcolor=blue shape=circle fontcolor=white]  
  
  center -> node1  
  center -> node2  
  center -> node3  
  
  node1 -> a  
  node1 -> b  
  node1 -> c  
  
  node2 -> d  
  node2 -> e  
  node2 -> f  
  
  node3 -> g  
  node3 -> h  
  node3 -> i  
}
```

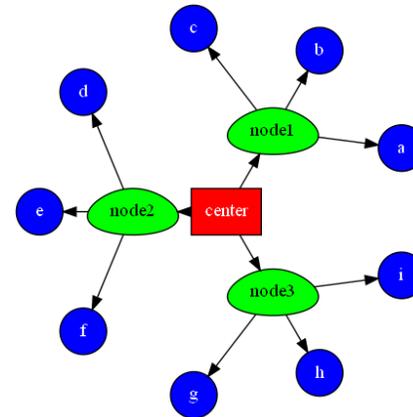


Алгоритмы визуализации графа (2)

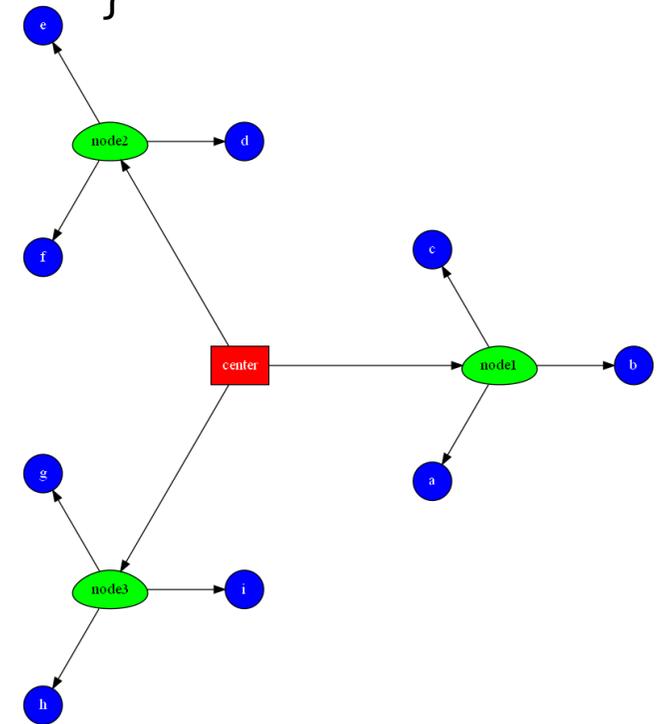
```
digraph {  
  graph [layout=dot]  
  ...  
}
```



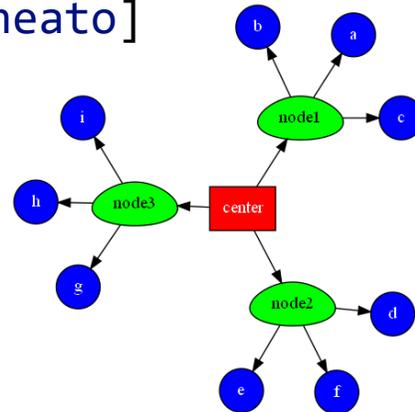
```
digraph {  
  graph [layout=twopi]  
  ...  
}
```



```
digraph {  
  graph [layout=circo]  
  ...  
}
```

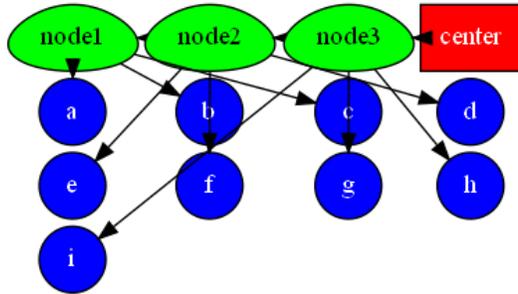


```
digraph {  
  graph [layout=neato]  
  ...  
}
```



Алгоритмы визуализации графа (3)

```
digraph {  
  graph [layout=osage]  
  ...  
}
```



```
digraph {  
  graph [layout=patchwork]  
  ...  
}
```

center	node1	node2	node3
a	d	e	
b	f	h	
c	g	i	

```
digraph {  
  graph [layout=fdp]  
  ...  
}
```

