

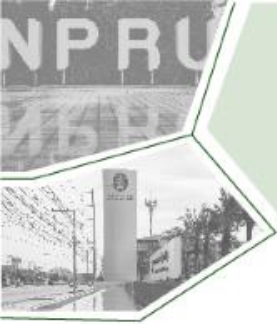


ทฤษฎีกราฟต้นไม้

ผู้ช่วยศาสตราจารย์สมเกียรติ ช่อเหมือน (tko@webmail.npru.ac.th)

สาขาวิชาวิศวกรรมซอฟต์แวร์ คณะวิทยาศาสตร์และเทคโนโลยี

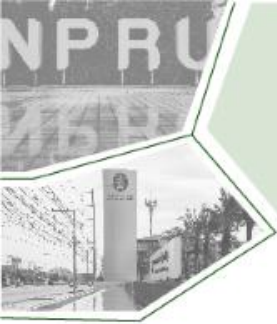




เนื้อหาการเรียนรู้

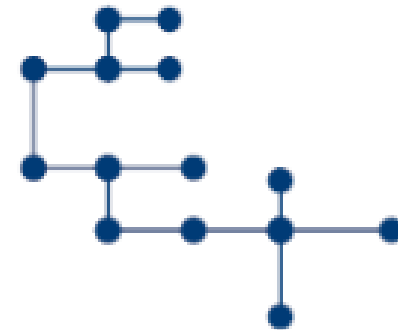
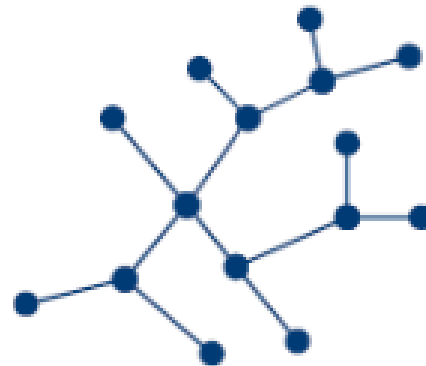
- กราฟต้นไม้
- ส่วนประกอบของกราฟต้นไม้
- ประเภทกราฟต้นไม้
- ลักษณะเฉพาะของต้นไม้
- แบบจำลองต้นไม้

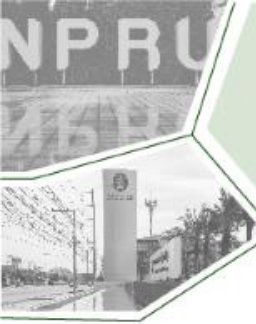




กราฟต้นไม้

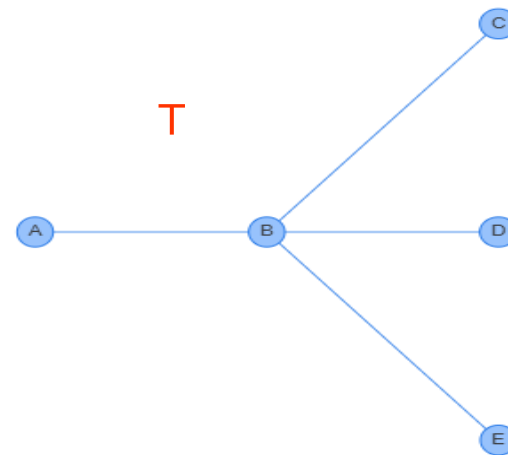
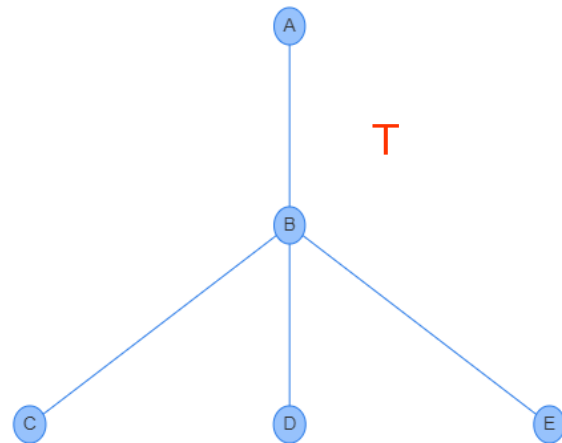
- กราฟต้นไม้ เป็นกราฟแบบต่อเนื่องรูปแบบหนึ่ง
- การกำหนดระดับชั้น ช่วยให้เห็นและเข้าใจความสัมพันธ์และแก้ปัญหา
- การแบ่งกลุ่มความสัมพันธ์ของกราฟ
- การประยุกต์ทางด้านต่าง ๆ
 - โครงสร้างข้อมูลทางคอมพิวเตอร์
 - โครงสร้างทางเคมีของสารประกอบไฮโดรคาร์บอน
 - การออกแบบวงจรไฟฟ้าและอิเล็กทรอนิกส์

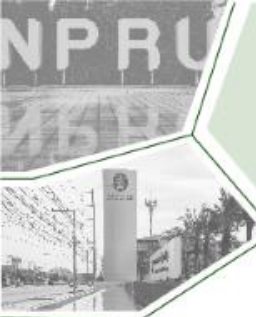




กราฟต้นไม้

- ต้นไม้ (tree) คือ กราฟที่สองจุดยอดใด ๆ มีวิถีเดินถึงกันได้เพียงวิถีเดียว หรือ เป็นกราฟที่ไม่
มีวัฏจักร แต่มีการเชื่อมต่อกัน
- กราฟที่ไม่เชื่อมต่อกันหมดเรียกว่า ป่า (forest)





จาวาสคริปต์ไลบรารี vis.js สำหรับแสดงกราฟต้นไม้

- การเขียนโปรแกรมด้วยภาษาจาวาสคริปต์

ส่วนที่ 1 การแสดงผลกราฟและการเรียกใช้ vis.js

```
<!doctype html>
<html>
<head>
  <title>Network</title>
  <script src="https://unpkg.com/vis-
network/standalone/umd/vis-network.min.js">
</script>
  <style type="text/css">
    #mynetwork {
      width: 600px;
      height: 400px;
      border: 1px solid lightgray;
    }
  </style>
</head>
<body>
  <div id="mynetwork"></div>
</body>
</html>
```

ส่วนที่ 2 การจัดเตรียมชุดข้อมูล

```
<script type="text/javascript">
  // create an array with nodes
  var nodes = new vis.DataSet([
    {id: 1, label: 'A'},
    {id: 2, label: 'B'},
    {id: 3, label: 'C'},
    {id: 4, label: 'D'},
    {id: 5, label: 'E'}
  ]);
  // create an array with edges
  var edges = new vis.DataSet([
    {from: 1, to: 2},
    {from: 2, to: 3},
    {from: 2, to: 4},
    {from: 2, to: 5}
  ]);
</script>
```

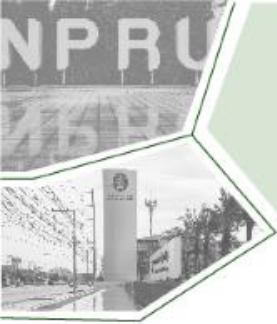
ส่วนที่ 3 การกำหนดโครงสร้างและจัดวางระดับชั้นของจุดยอด ++++++

```
var options = {
  layout: {
    autoResize: true,
    hierarchical: {
      direction: "UD", //"LR"
      sortMethod: "directed"
    }
  }
};
```



ส่วนที่ 4 การสร้างกราฟ

```
<script type="text/javascript">
  // create a network
  var container = document.getElementById('mynetwork');
  var data = {
    nodes: nodes,
    edges: edges
  };
  var options = {}; //ส่วนที่ 3
  var network = new vis.Network(container, data, options);
</script>
```



ส่วนประกอบของกราฟต้นไม้

- การเติบโตและแตกกิ่งก้านออกไปเป็นต้นไม้ขนาดใหญ่ หรือการแยกต้นไม้มีองค์ประกอบดังนี้

ใบ (leaf) จุดยอดที่มีระดับชั้นหรือดีกรีเท่ากับ 1

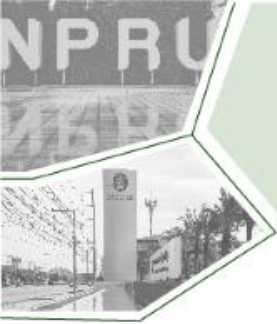
กิ่ง เส้นเชื่อมที่เชื่อมต่อใบ

ราก (root) จุดยอดใดจุดยอดหนึ่งที่ถูกกำหนดขึ้นมาให้เป็นราก

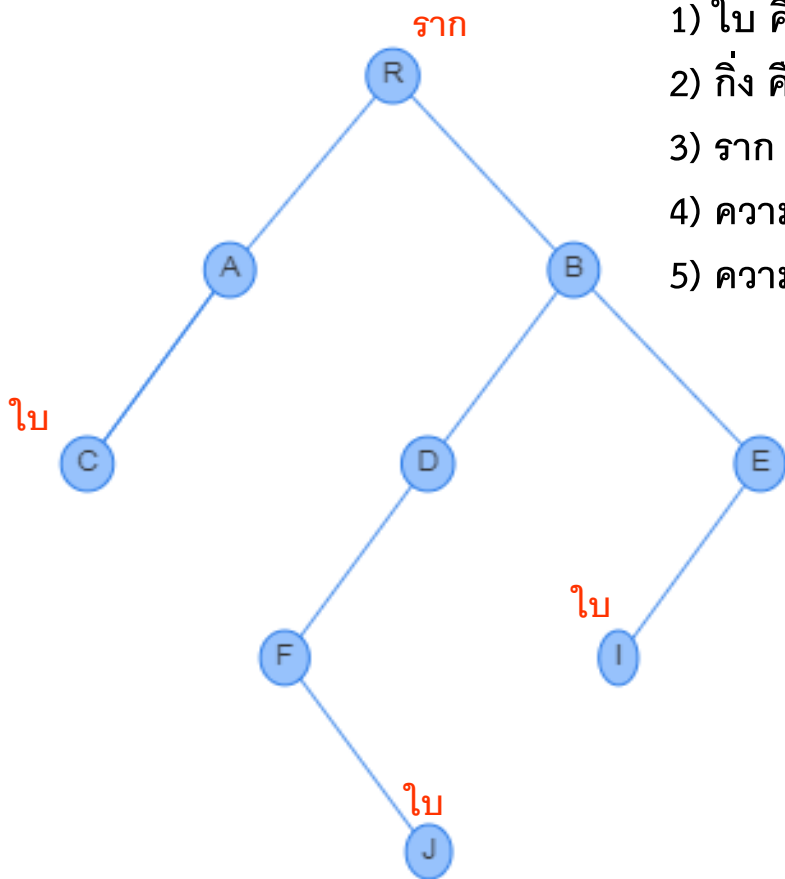
ความสูงของจุดยอด (vertices height) จำนวนเส้นเชื่อมบนวิถีจากจุดยอดใด ๆ ถึงราก

ความสูงของต้นไม้ (tree height) หมายถึง ความสูงของใบที่มากที่สุด

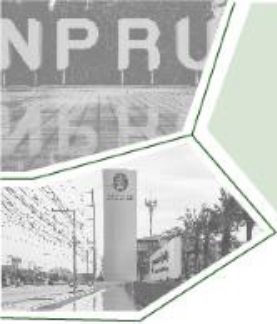




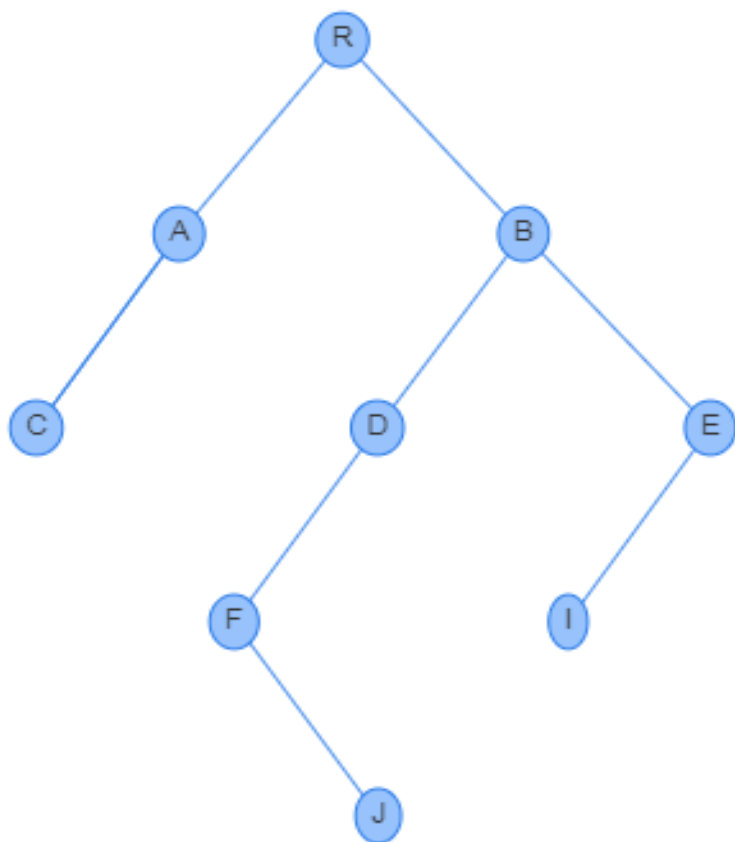
องค์ประกอบกราฟต้นไม้



- 1) ใบ คือ โหนด C, J และ I ซึ่งมีค่าดีกรีเท่ากับ 1
- 2) กิ่ง คือ เส้นเชื่อมทั้งหมด
- 3) ราก คือ โหนด R
- 4) ความสูงของจุด เช่น นับเส้นเชื่อมตั้งแต่โหนด R ถึง โหนด E เท่ากับ 2 เส้นเชื่อม
- 5) ความสูงของต้นไม้จากราก (R) ถึงใบที่อยู่ระดับมากที่สุดคือ โหนด J เท่ากับ 5 ระดับชั้น



เซต $T(V,E)$

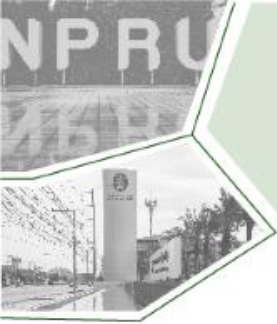


การเขียนโปรแกรมวาดส่วนประกอบของกราฟต้นไม้

```
// create an array with nodes
var nodes = new vis.DataSet([
  {id: 0, label : 'R' ,level:1},
  {id: 1, label : 'A' ,level:2}, {id: 2, label : 'B' ,level:2},
  {id: 3, label : 'C' ,level:3}, {id: 4 , label : 'D' ,level:3}, {id: 5, label : 'E' ,level:3},
  {id: 6, label : 'F' ,level:4}, {id: 7, label : 'I',level:4},
  {id: 8 , label : 'J' ,level:5}
]);

// create an array with edges
var edges = new vis.DataSet([
  {from: 0, to: 1},{from: 0, to: 2},
  {from: 1, to: 3},{from: 1, to: 3},{from: 2, to: 4},{from: 2, to: 5},
  {from: 4, to: 6},{from: 5, to: 7},
  {from: 6, to: 8}
]);
```





ประเภทกราฟต้นไม้

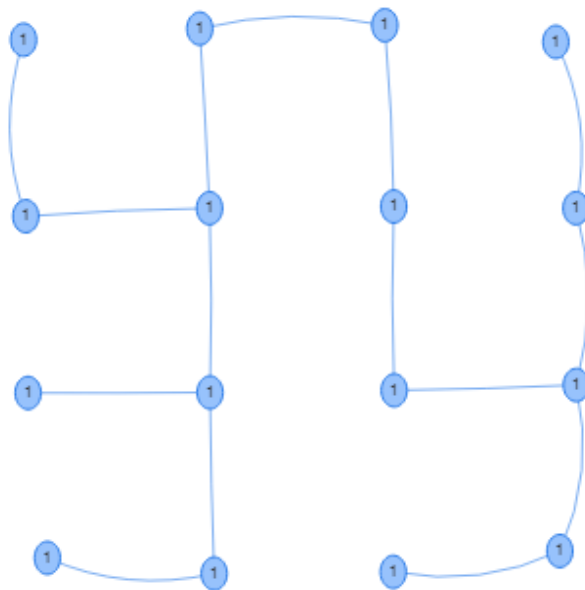
- ตัวบ่งบอกลักษณะเฉพาะของการจัดวางจุดยอดและการวาดเส้นเชื่อมต่อ
- การใช้งานหรือแก้ปัญหาในมุมมองที่แตกต่างกัน สามารถแยกกราฟต้นไม้เป็น 3 ประเภท
 - 1) ต้นไม้ทอดข้าม (spanning tree)
 - 2) ต้นไม้มีราก (root tree)
 - 3) ต้นไม้ย่อย (sub tree)

<https://mujergenial.net/>

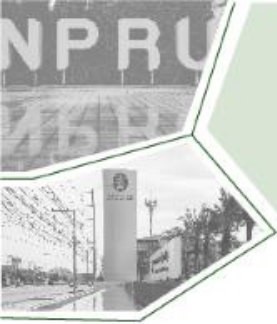


- **ต้นไม้ทอดข้าม (spanning tree)** หมายถึง กราฟย่อยของกราฟใด ๆ ซึ่งมีลักษณะเป็นต้นไม้ และมีจุดยอดทุกจุดของกราฟเป็นจุดยอดทุกจุดของต้นไม้ด้วย

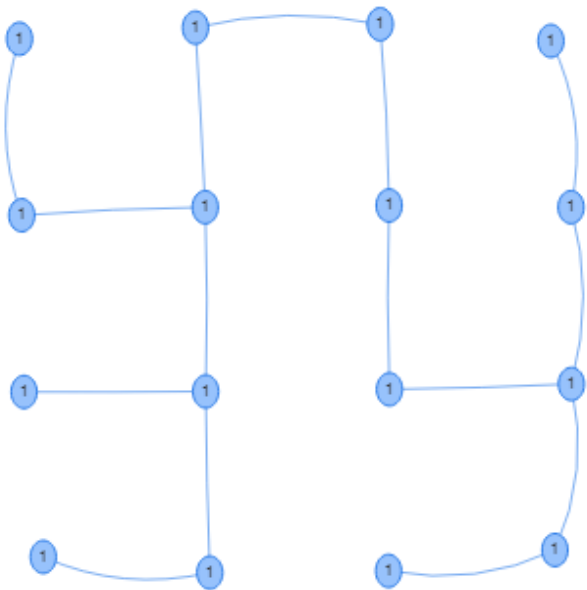
การเชื่อมต่อกันเป็นแถว 4x4 และจัดวางเป็นแนวเดียวกัน



กราฟต้นไม้แบบทอดข้ามเป็นการเชื่อมต่อกันของทุกจุดยอด



กราฟต้นไม้แบบทอดข้าม



การบล็อกพอร์ต (block port)

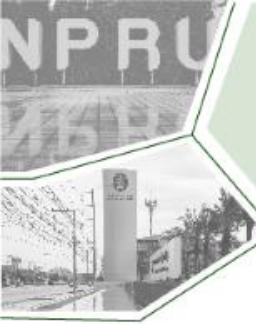
กราฟต้นไม้แบบทอดข้าม

```
// create an array with nodes
var nodes = new vis.DataSet([
  {id: 1, label : '1' }, {id: 2, label : '1' }, {id: 3, label : '1' }, {id: 4, label : '1' },
  {id: 5 , label : '1' }, {id: 6, label : '1' }, {id: 7, label : '1' }, {id: 8, label : '1' },
  {id: 9 , label : '1' }, {id: 10, label : '1' }, {id: 11, label : '1' }, {id: 12, label : '1' },
  {id: 13 , label : '1' }, {id: 14, label : '1' }, {id: 15, label : '1' }, {id: 16, label : '1' }
]);

// create an array with edges
var edges = new vis.DataSet([
  {from: 1, to: 2,color:'white'}, {from: 2, to: 3,color:'white'}, {from: 3, to: 4},
  {from: 5, to: 6}, {from: 6, to: 7}, {from: 7, to: 8},
  {from: 9, to: 10,color:'white'}, {from: 10, to: 11}, {from: 11, to: 12},
  {from: 13, to: 14}, {from: 14, to: 15}, {from: 15, to: 16},
  {from: 1, to: 5}, {from: 5, to: 9,color:'white'}, {from: 9, to: 13},
  {from: 2, to: 6}, {from: 6, to: 10,color:'white'}, {from: 10, to: 14},
  {from: 3, to: 7}, {from: 7, to: 11,color:'white'}, {from: 11, to: 15,color:'white'},
  {from: 4, to: 8,color:'white'}, {from: 8, to: 12}, {from: 12, to: 16,color:'white' }
]);
```

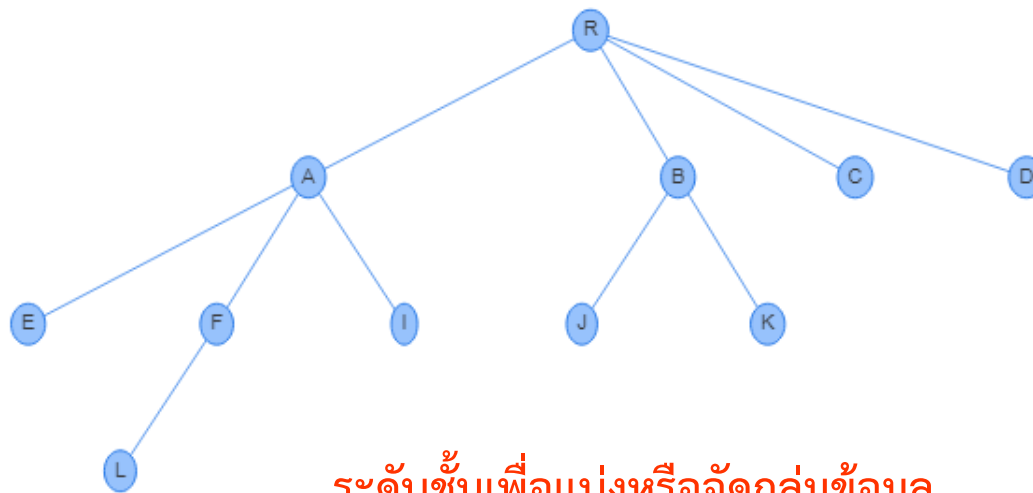


vis.js

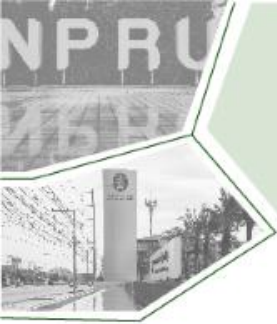


ต้นไม้มีราก

- **ต้นไม้มีราก (root tree)** เป็นต้นไม้ที่ถูกกำหนดให้ **จุดยอดหนึ่งจุด** ถูกกำหนดให้เป็น **ราก (root)** ซึ่งสามารถกำหนดทิศทางให้กับเส้นเชื่อมต่าง ๆ ได้อย่างเป็นธรรมชาติ โดยอาจเป็น **ทิศทางที่ชี้เข้าหาราก หรือ ออกจากราก** กำหนด (T, r) ที่มี r เป็นราก

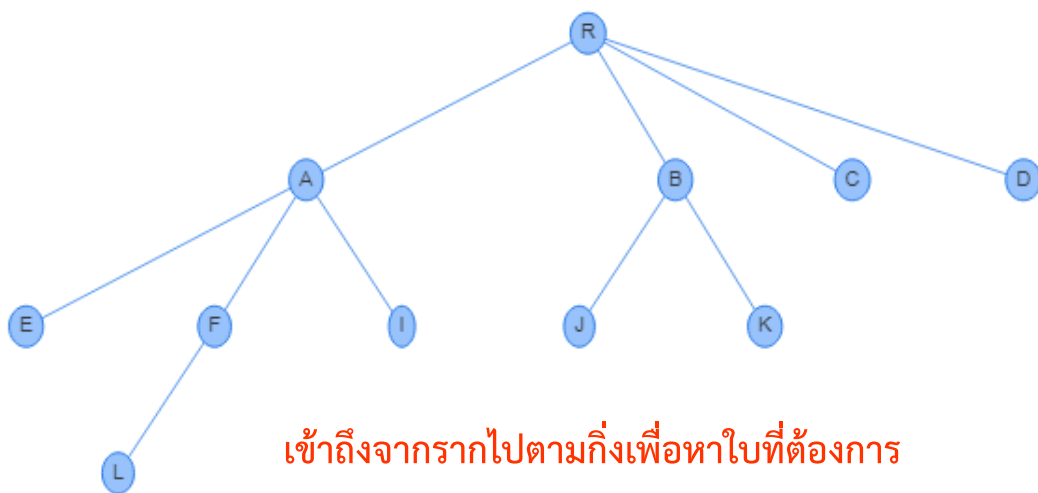


ระดับชั้นเพื่อแบ่งหรือจัดกลุ่มข้อมูล



กราฟต้นไม้แบบมีราก

โครงสร้างและจุดยอดแบบระดับชั้นของกราฟต้นไม้



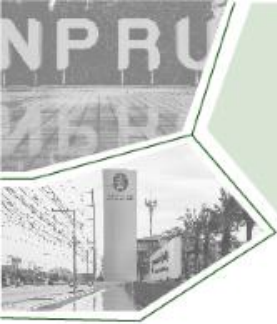
เข้าถึงจากรากไปตามกิ่งเพื่อหาใบที่ต้องการ

ตัวอย่างโค้ดการกำหนดเซตกราฟต้นไม้แบบมีราก

```
// create an array with nodes
var nodes = new vis.DataSet([
  {id: 0, label : 'R', level:1},
  {id: 1, label : 'A', level:2}, {id: 2, label : 'B', level:2},
  {id: 3, label : 'C', level:2}, {id: 4, label : 'D', level:2},
  {id: 5, label : 'E', level:3}, {id: 6, label : 'F', level:3},{id: 7, label : 'I', level:3},
  {id: 8, label : 'J', level:3}, {id: 9, label : 'K', level:3},
  {id: 10, label : 'L', level:4}
]);

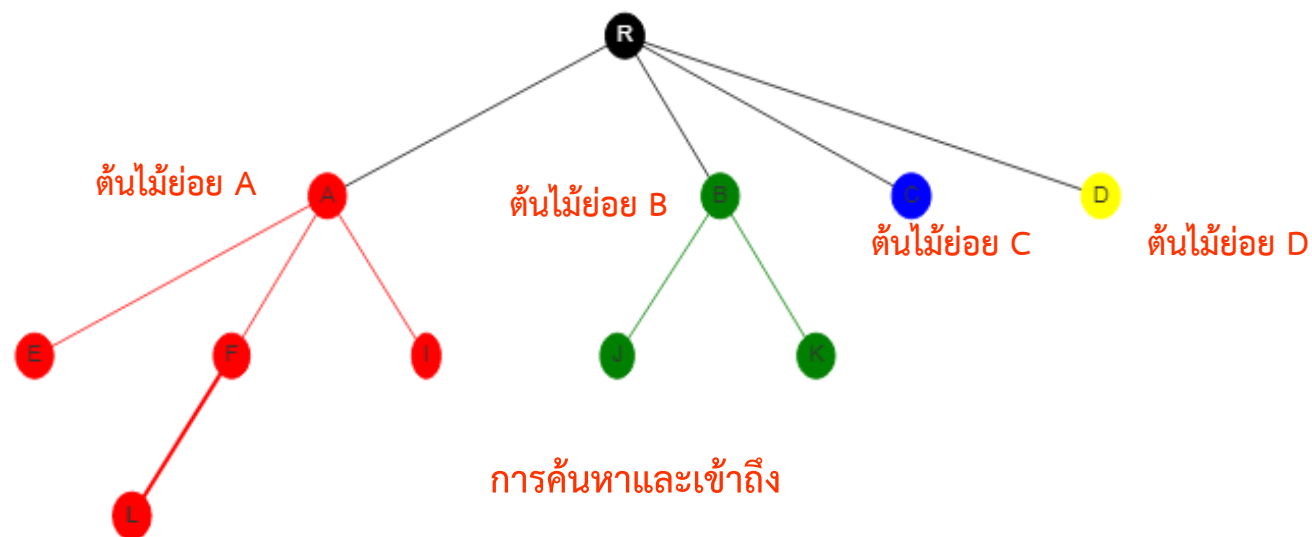
// create an array with edges
var edges = new vis.DataSet([
  {from: 0, to: 1},{from: 0, to: 2},{from: 0, to: 3},{from: 0, to: 4},
  {from: 1, to: 5},{from: 1, to: 6},{from: 1, to: 7},
  {from: 2, to: 8},{from: 2, to: 9},
  {from: 6, to: 10}
]);
```

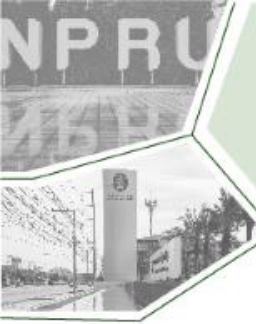




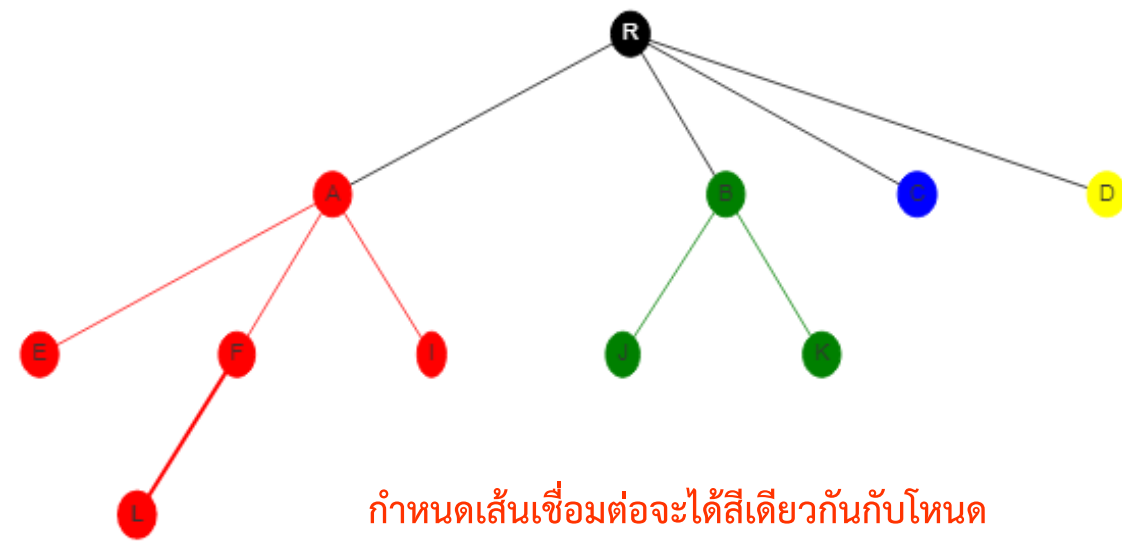
ต้นไม้ย่อย

- **ต้นไม้ย่อย (sub tree)** หมายถึงกราฟย่อยของต้นไม้ เป็น**ส่วนย่อยที่ต่อจากจุดยอด**ที่สนใจ ทำให้เห็นโครงสร้างหรือกลุ่มที่สนใจว่ามี**ความสัมพันธ์ที่แตกต่างกัน**ของแต่ละกลุ่ม





กราฟต้นไม้และการแบ่งต้นไม้ย่อย



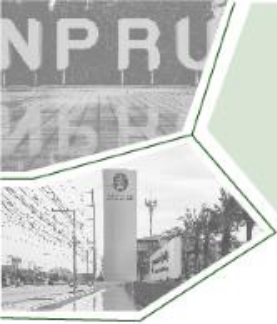
กำหนดเส้นเชื่อมต่อจะได้สีเดียวกันกับโหนด

กราฟต้นไม้และการแบ่งต้นไม้ย่อย

```
// create an array with nodes
var nodes = new vis.DataSet([
  {id: 0, label : 'R' ,level:0, color:'black',font:{color:'white'}},
  {id: 1, label : 'A' ,level:1 , color:'red'}, {id: 5, label : 'E' ,level:2, color:'red'},
  {id: 6, label : 'F' ,level:2, color:'red'}, {id: 10, label : 'L' ,level:3, color:'red'},
  {id: 7, label : 'I' ,level:2, color:'red'},
  {id: 2, label : 'B' ,level:1, color:'green'}, {id: 8 , label : 'J' ,level:2, color:'green'},
  {id: 9, label : 'K' ,level:2, color:'green'}
  {id: 3, label : 'C' ,level:1, color:'blue'},
  {id: 4 , label : 'D' ,level:1, color:'yellow'}
]);

// create an array with edges
var edges = new vis.DataSet([
  {from: 0, to: 1},{from: 0, to: 2},{from: 0, to: 3},{from: 0, to: 4},
  {from: 1, to: 5},{from: 1, to: 6},{from: 1, to: 7},
  {from: 2, to: 8},{from: 2, to: 9},
  {from: 6, to: 10}
]);
```



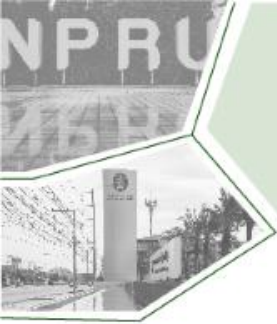


ลักษณะเฉพาะของต้นไม้

- ทฤษฎีบท

1. ให้ T เป็นกราฟที่ไม่มีวงวน กราฟ T เป็นต้นไม้ ก็ต่อเมื่อ จุดยอด 2 จุดใดๆ ใน T เชื่อมโยงกันได้ด้วยวิธีเพียงวิธีเดียว
2. ให้ T เป็นกราฟที่มีจำนวนจุดยอดเป็น n จุด กราฟ T เป็นต้นไม้ ก็ต่อเมื่อ กราฟ T ไม่มีวัฏจักร และมีเส้นเชื่อม $n - 1$ เส้น
3. ให้ T เป็นกราฟที่มีจำนวนจุดยอดเป็น n จุด กราฟ T เป็นต้นไม้ ก็ต่อเมื่อ กราฟ T เป็นกราฟเชื่อมโยงและมีเส้นเชื่อม $n - 1$ เส้น
4. ถ้า T เป็นต้นไม้ที่มีจุดยอดอย่างน้อย 2 จุด แล้วกราฟ T จะมีดีกรี 1 อย่างน้อย 2 จุด
5. ป่า (forest) ประกอบขึ้นจาก $T > 1$ ถ้า G เป็นกราฟแบบไม่มีทิศทางเชิงเดียวเรียกว่า ป่าได้ ก็ต่อเมื่อ G นั้นไม่มีวัฏจักรเชิงเดียว ดังนั้น ต้นไม้ต้นเดียวอาจเรียกว่า ป่า





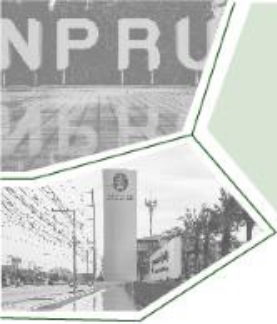
แบบจำลองต้นไม้

- โครงสร้างข้อมูลนามธรรมที่มีลักษณะการเรียงเป็นกิ่งก้านสาขาแตกแขนงออกไป ไม่มีวงวน
- สมาชิกจะถูกเก็บในประเภทข้อมูลชนิดวัตถุ (Object) หรือโครงสร้าง (Structure) ของโหนด (node)
- ตัวแปรเก็บตัวชี้ (Pointer) ไปยังโหนดอื่น ๆ
- ต้นไม้ในการจัดการข้อมูล
- การเปรียบเทียบกัน (comparable) ของค่าตัวเลขได้อย่างรวดเร็ว
- การเรียงลำดับความสำคัญของข้อมูล
 - การค้นหาในโครงสร้างต้นไม้
 - การจัดลำดับความสำคัญด้วยการสร้างตาราง
 - การจัดการค้นหาอย่างรวดเร็ว



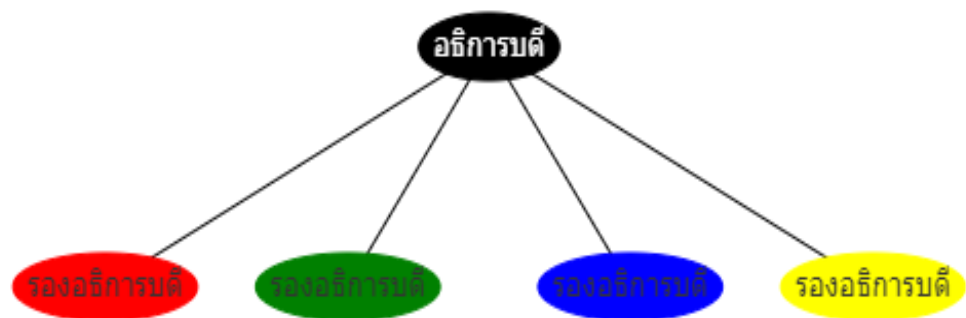
Object





แบบจำลองการบริหารงาน

การแบ่งตามหน้าที่และความรับผิดชอบ

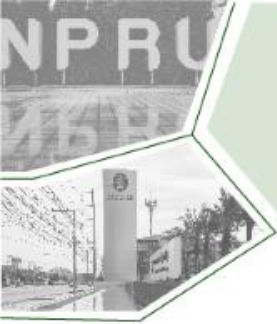


กราฟต้นไม้การบริหารงาน

```
var nodes = new vis.DataSet([
  {id: 0, label : 'อธิการบดี' ,level:0, color:'black',font:{color:'white'}}
  , {id: 1, label : 'รองอธิการบดี' ,level:1 , color:'red'}
  , {id: 2, label : 'รองอธิการบดี' ,level:1, color:'green'}
  , {id: 3, label : 'รองอธิการบดี' ,level:1, color:'blue'}
  , {id: 4 , label : 'รองอธิการบดี' ,level:1, color:'yellow'}
]);

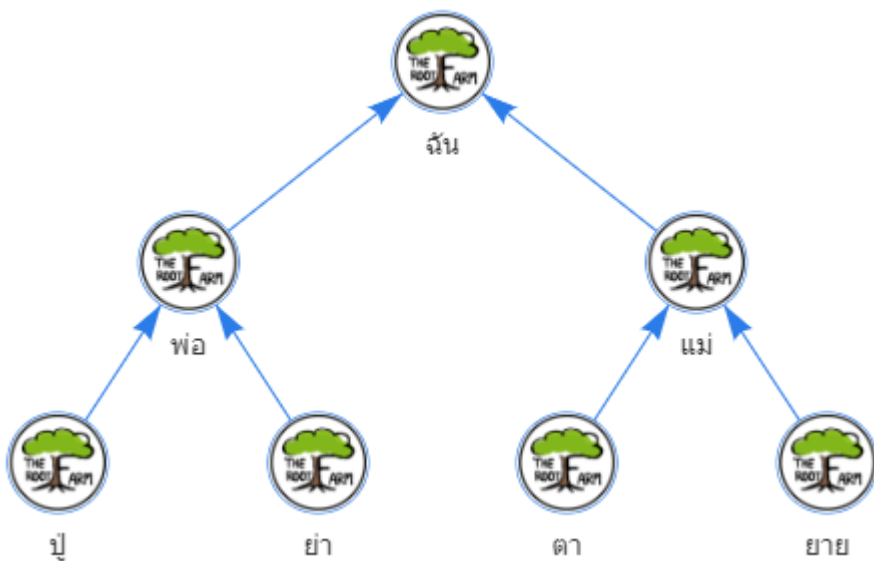
var edges = new vis.DataSet([
  {from: 0, to: 1},{from: 0, to: 2},{from: 0, to: 3},{from: 0, to: 4}
]);
```





แบบจำลองความสัมพันธ์ของครอบครัว

การสร้างความสัมพันธ์ของเครือญาติซึ่งมีระดับชั้น



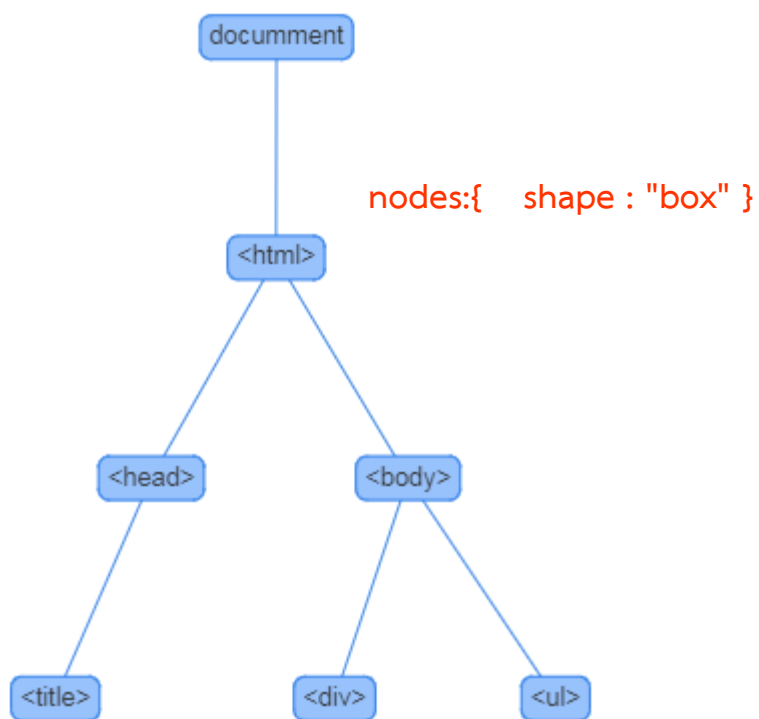
ต้นไม้แบบไบนารีทรี ทุกโหนดจะประกอบด้วย 2 ใบ

```
var DIR = 'img/';  
var nodes = new vis.DataSet([  
  {id: 0, label : 'จัน' ,level:0, shape: 'circularImage', image: DIR + 'p1.png'}  
  ,{id: 1, label : 'พ่อ' ,level:1, shape: 'circularImage', image: DIR + 'p1.png'}  
  ,{id: 2, label : 'แม่' ,level:1, shape: 'circularImage', image: DIR + 'p1.png'}  
  ,{id: 3, label : 'ปู่' ,level:2, shape: 'circularImage', image: DIR + 'p1.png'}  
  ,{id: 4, label : 'ย่า' ,level:2, shape: 'circularImage', image: DIR + 'p1.png'}  
  ,{id: 5, label : 'ตา' ,level:2, shape: 'circularImage', image: DIR + 'p1.png'}  
  ,{id: 6, label : 'ยาย' ,level:2, shape: 'circularImage', image: DIR + 'p1.png'}  
]);  
// create an array with edges  
var edges = new vis.DataSet([  
  {from: 0, to: 1 , arrows: 'from'}, {from: 0, to: 2, arrows: 'from'}  
  ,{from: 1, to: 3, arrows: 'from'}, {from: 1, to: 4, arrows: 'from'}  
  ,{from: 2, to: 5, arrows: 'from'}, {from: 2, to: 6, arrows: 'from'}  
]);
```



แบบจำลองโครงสร้างเอกสารเว็บ

ความสัมพันธ์ของออบเจ็กต์ในเอกสาร



กราฟต้นไม้โครงสร้างเอกสารเว็บ

```
var nodes = new vis.DataSet([
  {id: 0, label : 'document' ,level:0},
  {id: 1, label : '<html>' ,level:1},
  {id: 2, label : '<head>' ,level:2}, {id: 3, label : '<body>' ,level:2},
  {id: 4, label : '<title>' ,level:3}, {id: 5, label : '<div>' ,level:3},
  {id: 6, label : '<ul>' ,level:3},
]);
// create an array with edges
var edges = new vis.DataSet([
  {from: 0, to: 1},{from: 1, to: 2},{from: 1, to: 3},
  {from: 2, to: 4},{from: 3, to: 5},{from: 3, to: 6},
]);
```



แบบจำลองแผนผังเว็บไซต์

การเข้าถึงเนื้อหาในเว็บไซต์ หรืออาจเพิ่มตำแหน่งเข้าถึงทรัพยากร



การแสดงผลสัญลักษณ์

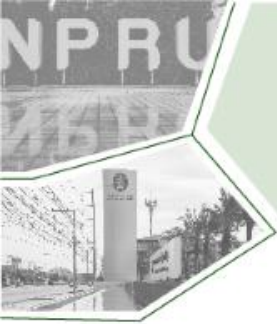
กราฟแผนผังเว็บไซต์

```
// create an array with nodes
var DIR = 'site/';
var nodes = new vis.DataSet([
  {id: 0, label : 'Homepage' ,level:1, shape: 'circularImage', image: DIR + 'home-icon.png'},
  {id: 1, label : 'Product' ,level:2, shape: 'circularImage', image: DIR + 'product.png'},
  {id: 2, label : 'Community' ,level:2, shape: 'circularImage', image: DIR + 'community.jpg'},
  {id: 3, label : 'About' ,level:2, shape: 'circularImage', image: DIR + 'about.png'},
  {id: 4, label : 'Overview' ,level:3, shape: 'circularImage', image: DIR + 'overview.jpg'},
  {id: 5, label : 'Shop' ,level:3, shape: 'circularImage', image: DIR + 'shop.png'},
  {id: 6, label : 'Blog' ,level:3, shape: 'circularImage', image: DIR + 'blog.png'},
  {id: 7, label : 'Guestbook' ,level:3, shape: 'circularImage', image: DIR + 'book.png'},
  {id: 8, label : 'Gallery' ,level:3, shape: 'circularImage', image: DIR + 'gallery.jpg'},
  {id: 9, label : 'Contact' ,level:3, shape: 'circularImage', image: DIR + 'contact.jpg'},
]);
// create an array with edges
var edges = new vis.DataSet([
  {from: 0, to: 1},{from: 0, to: 2},{from: 0, to: 3},
  {from: 1, to: 4},{from: 1, to: 5},
  {from: 2, to: 6},{from: 2, to: 7},
  {from: 3, to: 8},{from: 3, to: 9},
]);
```



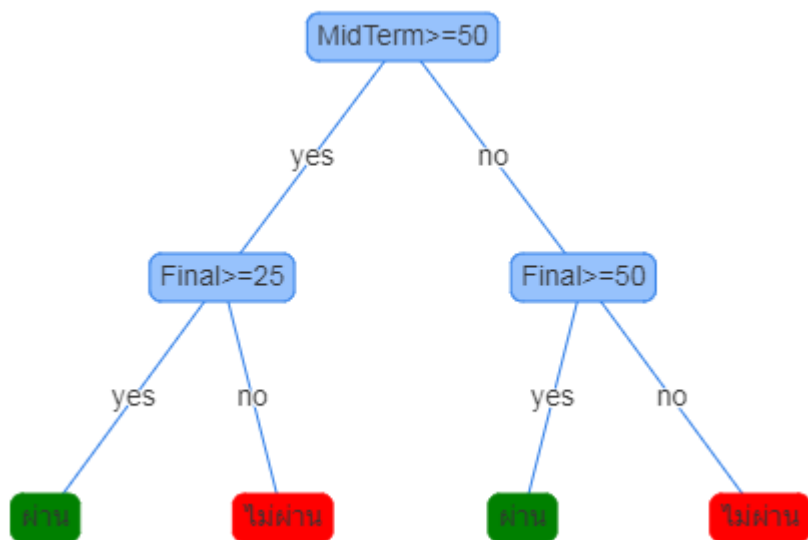
vis.js





แบบจำลองต้นไม้ในการตัดสินใจ

กำหนดเงื่อนไขและตัดสินใจ



คะแนนสอบกลางภาคและคะแนนสอบปลายภาค

กราฟต้นไม้แสดงการตัดสินใจในการสอบผ่าน

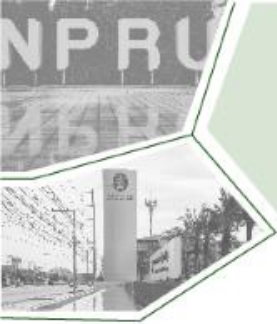
```
// create an array with nodes
var nodes = new vis.DataSet([
  {id: 0, label : 'MidTerm>=50' , level:0},
  {id: 1, label : 'Final>=25' , level:1}, {id: 2, label : 'Final>=50' ,level:1},
  {id: 3, label : 'ผ่าน' , level:2,color:'green'}, {id: 4, label : "ไม่ผ่าน" , level:2,color:'red'},
  {id: 5, label : 'ผ่าน' , level:2,color:'green'}, {id: 6, label : "ไม่ผ่าน" , level:2,color:'red'},
]);

// create an array with edges
var edges = new vis.DataSet([
  {from: 0, to: 1,label:'yes'}, {from: 0, to: 2,label:'no'},
  {from: 1, to: 3,label:'yes'}, {from: 1, to: 4,label:'no'},
  {from: 2, to: 5,label:'yes'}, {from: 2, to: 6,label:'no'},
]);
```



vis.js





สรุปท้ายบท

- กราฟเชื่อมโยงประเภทหนึ่งที่นิยมนำมาใช้ในการจัดกลุ่มหรือระดับชั้นจุดยอด
- ไม่มีการเกิดวัฏจักรในกราฟ ส่วนใหญ่เริ่มต้นจากรากแตกแยกออกเป็นกิ่งและใบ
- ระบุระดับชั้นหรือความสูงได้จากการนับจากกราฟและจำนวนใบที่เกิดขึ้นได้ของต้นไม้
- กราฟต้นไม้แบ่งได้เป็น 1) ต้นไม้ทอดข้าม 2) ต้นไม้มีราก 3) ต้นไม้ย่อย
- ตัวอย่างในการเขียนโปรแกรมสร้างแบบจำลองต้นไม้

