МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра: САП

Звіт до виконаної лабораторної роботи №5

з дисципліни “Дискретна математика”

на тему: "Основи теорії дерев"

Виконав:

студент групи ПП-16

Якіб’юк Ігор

Прийняв:

Асистент каф. САП
Іванина В. В.

Львів - 2023

Лабораторна робота № 5

Мета роботи: Мета роботи – ознайомитись на практиці із основними поняттями теорії дерев, вивчити можливості програми Maple18 для розв'язування задач про дерева.

Хід роботи:
Варіант 29

Завдання 4.1:

Завдання 4.1. Побудувати та відобразити у програмі Maple18 дерево, яке містить:

29.12 ребер

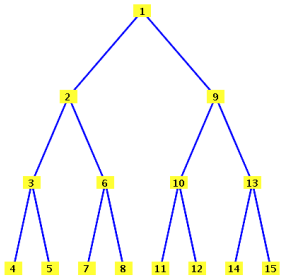
До кожного з побудованих дерев записати список сумiжностi та масив його елементів.

Розв'язання 4.1:

```
K := CompleteBinaryTree(3);
```

K := Graph 147: an undirected unweighted graph with 15 vertices and 14 edge(s)

```
DrawGraph(K);
```




```
NumberOfEdges(K);
```

14

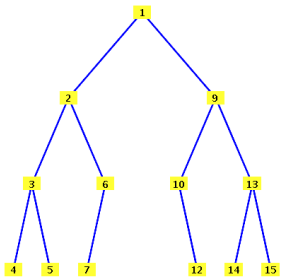
```
H := DeleteEdge(K, {{10, 11}, {6, 8}});
```

H := Graph 147: an undirected unweighted graph with 15 vertices and 12 edge(s)

```
H := DeleteVertex(K, [8, 11])
```

H := Graph 148: an undirected unweighted graph with 13 vertices and 12 edge(s)

```
DrawGraph(H);
```




```
IsTree(H);
```

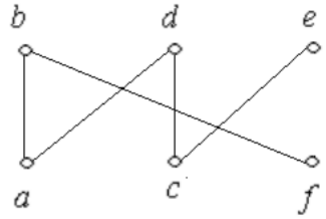
true

Батько	Сини
1	2, 9
2	3, 6
9	10, 13
3	4, 5
6	7
10	12
13	14, 15

0	1	2	3	4	5	6	7	8	9	10	11	12
1	2	9	3	6	10	13	4	5	7	12	14	15

Завдання 4.2:

Завдання 2. Користуючись програмними кодами (процедура *type()*), наведеними в теоретичній частині, перевірити чи зображений на рисунку граф є деревом.



29

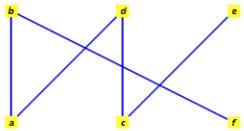
Розв'язання 4.2:

```
B := Graph({{a, b}, {a, d}, {b, f}, {c, d}, {c, e}});
```
B := Graph 149: an undirected unweighted graph with 6 vertices and 5 edge(s)

```
vp := [[1, 0], [1, 1], [2, 0], [2, 1], [3, 1], [3, 0]];
```
vp := [[1, 0], [1, 1], [2, 0], [2, 1], [3, 1], [3, 0]]

```
SetVertexPositions(B, vp);

DrawGraph(B);
```




```
`type/Tree` := proc(obj)
 local result;
 try
   result := GraphTheory[IsTree](obj);
 catch:
   result := false;
 end try;
 return result;
end proc:
type(B, Tree);
```
true

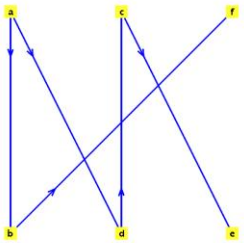
Згідно з результату 4.2, завдання 4.3.1:

4.3.1. Якщо заданий граф згідно з Вашим варіантом є деревом, то користуючись наведеними в теоретичній частині програмними кодами (процедури *type/RTree()* та *DrawRTree()*), перевірити чи породжений кореневий граф є деревом та нарисувати його, задавши попередньо його кореневу вершину (граф *firstRooted)*.

Розв'язання 4.3.1:

```
B := Graph({["a", "b"], ["a", "d"], ["b", "f"], ["d", "c"], ["c", "e"]});
```
B := Graph 4: a directed unweighted graph with 6 vertices and 5 arc(s)

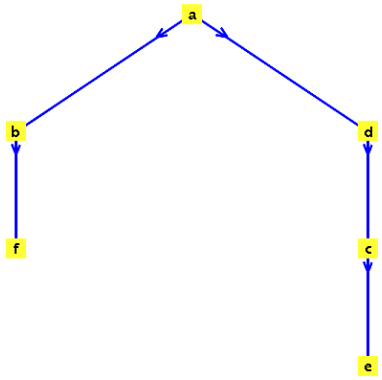
```
DrawGraph(B);
```




```
`type/RTree` := proc(obj) local R, Alist, v, i; uses GraphTheory; if not type(obj, Graph) then
    return false end if; if not IsTree(UnderlyingGraph(obj)) then return false; end if; R
    := GetGraphAttribute(obj, "root"); if not R in Vertices(obj) then return false; end if; Alist
    := [R]; i := 1; while i ≤ nops(Alist) do Alist := [op(Alist), op(Departures(obj,
    Alist[i]))]; i := i + 1; end do; if{op(Alist)} = {op(Vertices(obj))} then return true;
    else return false; end if; end proc:
SetGraphAttribute(B, "root" = "a");
GetGraphAttribute(B, "root");
```
"a"

```
type(B, RTree);
```
true

```
DrawRTree := proc(firstRooted :: RTree)
local R, U, P;
uses GraphTheory;
R := GetGraphAttribute(firstRooted, "root");
U := UnderlyingGraph(firstRooted);
DrawGraph(U, style = tree, root = R) :
P := GetVertexPositions(U, style = tree, root = R);
SetVertexPositions(firstRooted, P);
DrawGraph(firstRooted);
end proc:
DrawRTree(B)
```



Завдання 4.4:

Завдання 4.4. Для кореневого дерева із завдання 4.3, користуючись наведеними в теоретичній частині програмними кодами:

1) знайдіть предків вершини ***d*** (*FindParent*);
2) знайдіть нащадків вершини **c** (*FindChildren*);
3) визначте список дітей дерева (нащадки кореневої вершини) (*FindChildren*);
4) встановіть чи вершина ***e*** є внутрішньою чи листком (*IsInternal, IsLeaf*);
5) визначте перелік всіх листків дерева (*FindLeaves*).

Розв'язання 4.4:

```
FindParent := proc(T :: RTree, v)
local A;
A := GraphTheory[Arrivals](T, v);
if nops(A) = 1 then
return op(A);
elif nops(A) = 0 then
return FAIL;
else
error "The given graph is not a tree.";
end if;
end proc:
FindParent(B, "d");
```

"a"

```
FindChildren := proc(T :: RTree, v)
return GraphTheory[Departures](T, v);
end proc:
FindChildren(B, "c");
```

["e"]

```
FindChildren(B, "a");
```

["b", "d"]

```
IsInternal := proc(T :: RTree, v)
if GraphTheory[Departures](T, v) ≠ [ ] then
return true;
else
return false;
end if;
end proc:
IsInternal(B, "e");
```

false

```
FindLeaves := proc(T :: RTree)
local Leaves, v;
uses GraphTheory;
Leaves := { };
for v in Vertices(T) do
if IsLeaf(T, v) then
Leaves := Leaves union {v};
end if;
end do;
return Leaves;
end proc:
FindLeaves(B);
```

```
IsLeaf := proc(T :: RTree, v)
if GraphTheory[Departures](T, v) = [ ] then
return true;
else
return false;
end if;
end proc:
IsLeaf(B, "e");
```

true

Завдання 4.5:

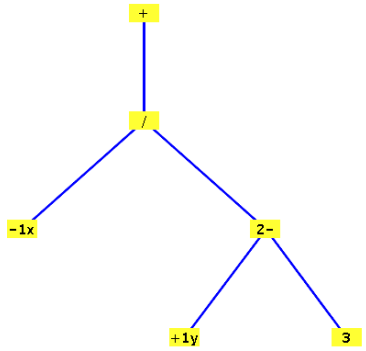
Завдання 4.5. Побудувати кореневі дерева, які відповідають математичним виразам

29. $x/(y-3)+x*((8*y-4)+x^2))$

Розв'язання 4.5:

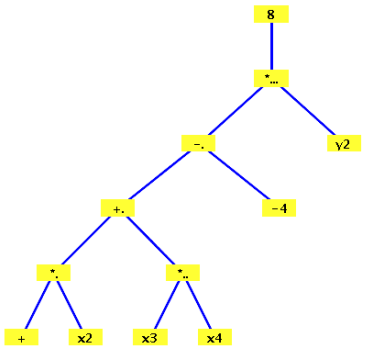
```
MathGraph1 := Graph({{"+", "/"}, {"/", "-1x"}, {"/", "2-"}, {"2-", "3"}, {"2-", "+1y"}});
  MathGraph1 := Graph 5: an undirected unweighted graph with 6 vertices and 5 edge(s)

DrawGraph(MathGraph1);
```



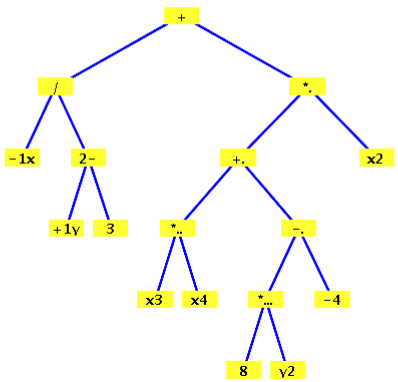

```
MathGraph2 := Graph({{"+", "·."}, {"·.", "x2"}, {"·.", "+."}, {"+.", "-."}, {"+.", "·.."},
    {"·..", "x3"}, {"·..", "x4"}, {"-.", "·..."}, {"·...", 8}, {"·...", "y2"}, {"-.", "-4"}});
MathGraph2 := Graph 6: an undirected unweighted graph with 12 vertices and 11 edge(s)

SetGraphAttribute(MathGraph1, "root" = "+");
DrawGraph(MathGraph2);
```




```
FnGraph := GraphUnion(MathGraph1, MathGraph2);
  FnGraph := Graph 7: an undirected unweighted graph with 17 vertices and 16 edge(s)

DrawGraph(FnGraph, style = tree, root = "+");
```



В останньому завданні побудував кореневе дерево, що відповідає математичному виразу, однак через особливості Maple 18 деяким операндам були надані індекси ('.', '+1', '-1', '2', '3', '4').

Висновок: На даній лабораторній роботі я ознайомився з основними поняттями теорії дерев, а також навчився розв’язувати з ними задачі у програмному середовищі Maple18. Під час виконання роботи: я намалював дерево з 12 ребрами, дослідив чи наданий мені граф є деревом. Пізніше за допомогою функцій в теоретичному матеріалі, встановив орієнтованому графу тип дерево і намалював його. Спочатку з цим виникли деякі проблеми, через те, що в 1 ребрі був неправильний напрям, але я її швидко пофіксив, і граф успішно був виведений на екран. Також уже намальований граф перевірив на функціях з теорії, а саме: FindParent, FindChildren, IsInternal, IsLeaf і FindLeaves. І в останньому завданні побудував кореневе дерево, що відповідає математичному виразу, однак через особливості Maple 18 деяким операндам були надані індекси (‘.’, ‘+1’, ‘-1’, ‘2’, ‘3’, ‘4’).