Лабораторна робота №6.
Бінарні дерева.

Мета роботи: Вивчити способи ефективного зберігання та обробки інформації на прикладі бінарних дерев.

Задача роботи: Написати програму роботи з бінарними деревами пошуку.

Теоретичні відомості.

Дерево – це сукупність елементів, що називаються вузлами (один з яких корінь), та відношень („батьківських”), що утворюють ієрархічну структуру вузлів. Вузли можуть бути елементами будь-якого типу (літерами, рядками, числами). До основних операцій з деревами належать: внесення елемента в дерево, обхід дерев та вилучення елемента з дерева. Під *обходом бінарного дерева* розуміють визначений порядок проходження усіх вершин дерева. Розрізняють декілька способів обходу дерев: прямий, зворотній та симетричний порядки обходу.

1. *Прямий порядок обходу* бінарного дерева можна визначити в такий спосіб (рис. 7): потрапити в корінь R; пройти в прямому порядку ліве піддерево A; пройти в прямому порядку праве піддерево B.

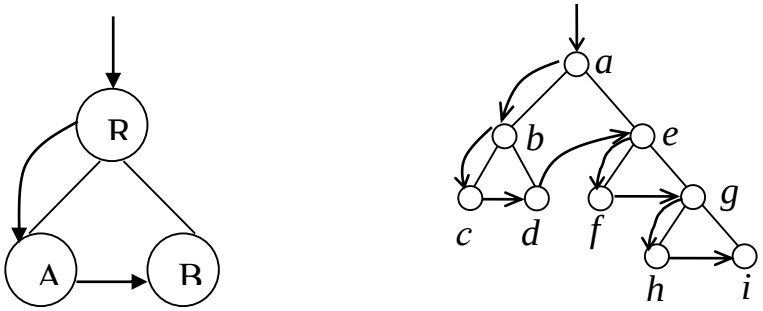

Рис. 7. Прямий порядок обходу бінарного дерева

2. Обхід бінарного дерева в *зворотному порядку* можна визначити в аналогічній формі (рис. 8): пройти в зворотному порядку ліве піддерево A; пройти в зворотному порядку праве піддерево B; потрапити в корінь R.

3. Визначимо ще один порядок обходу бінарного дерева, називаний *симетричним* (рис. 9): пройти в симетричному порядку ліве піддерево A; потрапити в корінь R; пройти в симетричному порядку праве піддерево B.

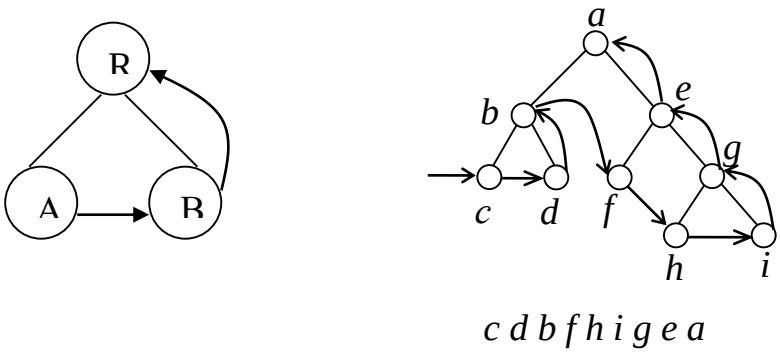

Рис. 8. Зворотний порядок обходу бінарного дерева

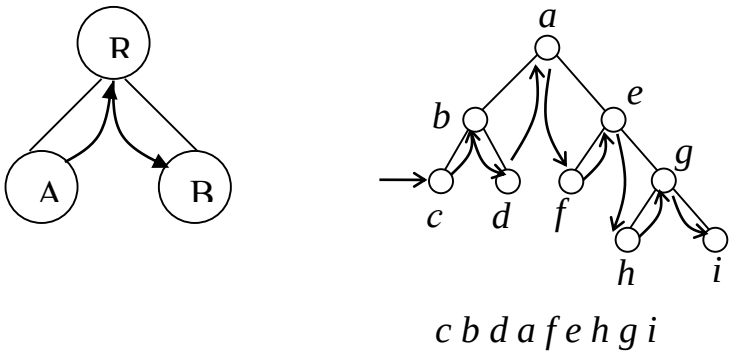

Рис. 9. Симетричний порядок обходу бінарного дерева

Розглянемо реалізацію дерева за допомогою покажчиків. Об’явимо дерево як:

```
Type
  TreeLink = ^Tree;
  Tree = Record
      Data : <тип даних>;
      Left, Right : TreeLink;
  End;
```

Корінь дерева опишемо в розділі опису змінних:

```
Var   kd : TreeLink;
```

Опишемо процедуру включення в дерево нового вузла. При включенні в дерево вузол вставляється або як піддерево вже існуючого вузла або як єдиний вузол дерева. Тому й лівий і правий зв'язки нового вузла повинні бути Nil. Коли дерево порожнє, значення передане у вигляді параметра покажчика дорівнює Nil. У цьому випадку потрібно змінити його так, щоб він вказував на новий вузол, що був включений як кореневий. При включенні наступного елемента переданий з основної програми параметр *t* уже не буде дорівнює nil, і треба приймати рішення щодо того, в яке піддерево необхідно включити новий вузол.

```
Procedure InsTree(n : integer; var t : TreeLink);
 Begin
  if t = nil     then
     begin new(t);
       with t^ do
         begin
           Left := nil;
           Right := nil;
           Data := n;
         end;
     end;
    else
     if n<=t^.data   then
        InsTree(n, t^.Left)
      else
        InsTree(n, t^.Right)
 End;
```

Опишемо процедуру виводу значень елементів бінарного дерева на екран. Для цього необхідно виконати повний обхід дерева. При обході дерева його окремі вузли відвідуються в окремому порядку: прямому, зворотному або симетричному. Процедура виводу дерева при зворотному обході має такий вигляд:

```
Procedure PrintTree(t : TreeLink);
 Begin
  if t<>nil
    then
     begin
       PrintTree(t^.Left);
       Write(t^.Data:3);
       PrintTree(t^.Right);
     end;
 End;
```

Основна програма здійснює введення чисел з клавіатури. Використаються: змінна nd типу TreeLink - значення покажчика на корінь дерева; змінна Digit типу integer для зберігання чергового введеного числа.

```
Begin
 writeln('Введіть вузли дерева. Закінчення введення – 0');
  kd := nil;
  read(Digit);
  while Digit <> 0 do
    begin
      InsTree(Digit, kd);
      writeln(' Введіть чергове число ');
      read(Digit);
    end;  PrintTree(kd);
End.
```

Завдання.

1. Створити бінарне дерево пошуку (для кожного вузла дерева у всіх лівих вузлах повинні перебувати числа менші, а в правих більші, ніж числа, що зберігаються в цьому вузлі), елементи якого вводяться із клавіатури й мають цілий тип.
2. Здійснити обхід дерева в прямому, зворотньому та симетричному порядках.
3. Написати функцію, яка знаходить найбільший та найменший елементи дерева.
4. Написати процедуру, яка видаляє з дерева всі парні елементи.
5. Написати процедуру, яка визначає число входжень заданого елемента в дерево.
6. Написати функцію, яка підраховує суму всіх елементів дерева.

Контрольні запитання.

1. Дати визначення: дерево, бінарне дерево, корінь та листи дерева, впорядковані дерева, дерево пошуку, висота та глибина дерева.
2. Схеми обходу дерев.
3. Яким є принцип динамічної побудови дерева?
4. У чому полягає схожість та відмінність таких структур даних як зв’язний список, стек, дерево?