**LAPORAN**

**PRAKTIKUM STRUKTUR DATA**

**Pertemuan ke – 10**

****

Disusun Oleh :

Rendra Eka Herlambang

175410028

**LABORATORIUM TERPADU**

**SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER AKAKOM YOGYAKARTA**

**2018**

**POHON**

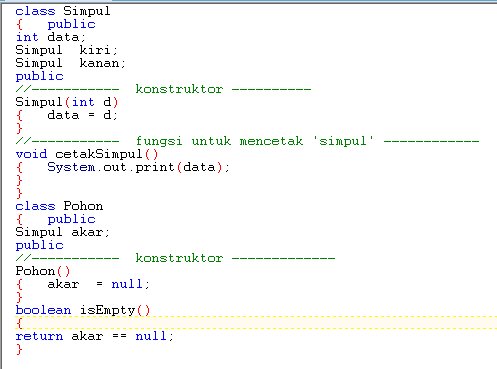
1. **TUJUAN**

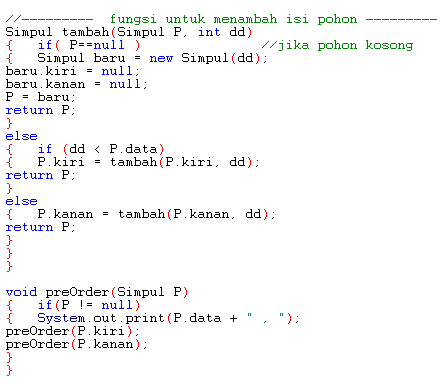
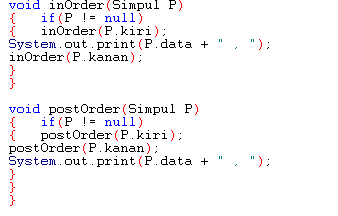
-Mahasiswa dapat membuat program double linked list

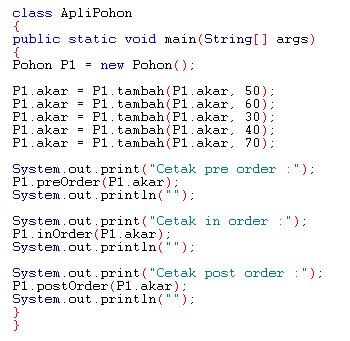
- Mahasiswa mampu mengimplementasikan double linked list.

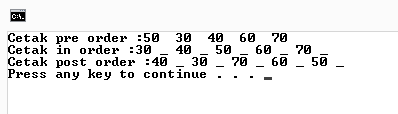
- Mahasiswa mampu mengimplemetasikan model senarai ganda.

1. **PEMBAHASAN PRAKTIK**







Pembahasan

class Simpul

{ public

int data;

Simpul kiri;

Simpul kanan;

public

//----------- konstruktor ----------

//yaitu mendekrasikan konstruktor pada simpul kanan dan kiri

Simpul(int d)

{ data = d;

}

//----------- fungsi untuk mencetak 'simpul' ------------

void cetakSimpul()

{ System.out.print(data);

}

}

//yaitu untuk mencetak simpul yaitu pada method cetak simpul.

class Pohon

{ public

Simpul akar;

Public

//----------- konstruktor -------------

Pohon()

{ akar = null;

}

boolean isEmpty()

{

return akar == null;

}

//yaitu untuk mendekrasikan class Pohon pada simpul akar,maka pohon jika akar=null maka kosong.

//--------- fungsi untuk menambah isi pohon ---------

Simpul tambah(Simpul P, int dd)

{ if( P==null ) //jika pohon kosong

{ Simpul baru = new Simpul(dd);

baru.kiri = null;

baru.kanan = null;

P = baru;

return P;

//yaitu untuk menambah isi pohon maka pada method simpul tambah ,jika p==null maka kosong,baru.kiri =null ,maka baru.kanan=null maka p=baru mak membalikan data.

}

else

{ if (dd < P.data)

{ P.kiri = tambah(P.kiri, dd);

return P;

}

else

{ P.kanan = tambah(P.kanan, dd);

return P;

}

}

}

//yaitu jika data < p.data maka p.kiri maka tambah data pada kiri,jika tidak maka p.kanan maka menambahkan data pada kanan.

void preOrder(Simpul P)

{ if(P != null)

{ System.out.print(P.data + " ");

preOrder(P.kiri);

preOrder(P.kanan);

}

}

//yaitu untuk method preOrder maka jika p tidak =null maka mencetak simpul pada kanan dan kiri.

void inOrder(Simpul P)

{ if(P != null)

{ inOrder(P.kiri);

System.out.print(P.data + " \_ ");

inOrder(P.kanan);

}

}

//maka jika p tidak null maka inOrder p.kiri maka inOrder kanan.

void postOrder(Simpul P)

{ if(P != null)

{ postOrder(P.kiri);

postOrder(P.kanan);

System.out.print(P.data + " \_ ");

}

}

}

//yaitu pada method postOrder,jika p=null maka posOrdernya kanan.

class ApliPohon

{

public static void main(String[] args)

{

Pohon P1 = new Pohon();

P1.akar = P1.tambah(P1.akar, 50);

P1.akar = P1.tambah(P1.akar, 60);

P1.akar = P1.tambah(P1.akar, 30);

P1.akar = P1.tambah(P1.akar, 40);

P1.akar = P1.tambah(P1.akar, 70);

//yaitu menjelaskan bahwa class Aplipohon maka pada method pohon maka isi akar 50,60,30,40,70.

System.out.print("Cetak pre order :");

P1.preOrder(P1.akar);

System.out.println("");

//yaitu untuk mencetak pada bagian preorder

System.out.print("Cetak in order :");

P1.inOrder(P1.akar);

System.out.println("");

//yaitu untuk mencetak bagian Cetak in order

System.out.print("Cetak post order :");

P1.postOrder(P1.akar);

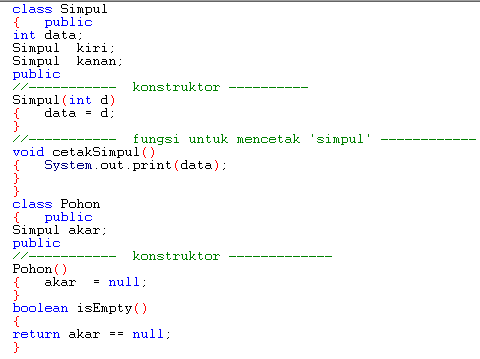
System.out.println("");

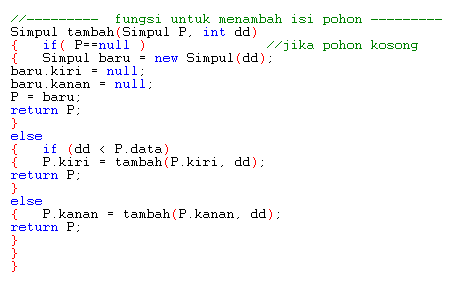
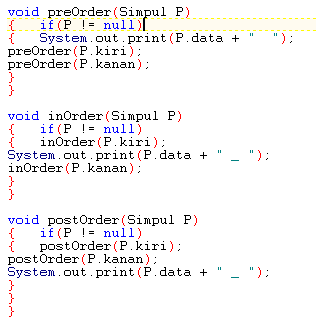
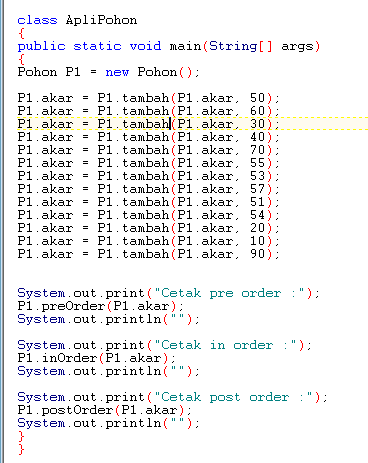
}

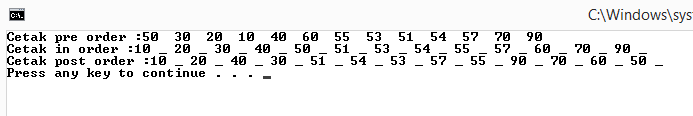
}

//yaitu untuk output cetak post order.

LATIHAN





Pembahasan

class Simpul

{ public

int data;

Simpul kiri;

Simpul kanan;

public

//----------- konstruktor ----------

//yaitu mendekrasikan konstruktor pada simpul kanan dan kiri

Simpul(int d)

{ data = d;

}

//----------- fungsi untuk mencetak 'simpul' ------------

void cetakSimpul()

{ System.out.print(data);

}

}

//yaitu untuk mencetak simpul yaitu pada method cetak simpul.

class Pohon

{ public

Simpul akar;

Public

//----------- konstruktor -------------

Pohon()

{ akar = null;

}

boolean isEmpty()

{

return akar == null;

}

//yaitu untuk mendekrasikan class Pohon pada simpul akar,maka pohon jika akar=null maka kosong.

//--------- fungsi untuk menambah isi pohon ---------

Simpul tambah(Simpul P, int dd)

{ if( P==null ) //jika pohon kosong

{ Simpul baru = new Simpul(dd);

baru.kiri = null;

baru.kanan = null;

P = baru;

return P;

//yaitu untuk menambah isi pohon maka pada method simpul tambah ,jika p==null maka kosong,baru.kiri =null ,maka baru.kanan=null maka p=baru mak membalikan data.

}

else

{ if (dd < P.data)

{ P.kiri = tambah(P.kiri, dd);

return P;

}

else

{ P.kanan = tambah(P.kanan, dd);

return P;

}

}

}

//yaitu jika data < p.data maka p.kiri maka tambah data pada kiri,jika tidak maka p.kanan maka menambahkan data pada kanan.

void preOrder(Simpul P)

{ if(P != null)

{ System.out.print(P.data + " ");

preOrder(P.kiri);

preOrder(P.kanan);

}

}

//yaitu untuk method preOrder maka jika p tidak =null maka mencetak simpul pada kanan dan kiri.

void inOrder(Simpul P)

{ if(P != null)

{ inOrder(P.kiri);

System.out.print(P.data + " \_ ");

inOrder(P.kanan);

}

}

//maka jika p tidak null maka inOrder p.kiri maka inOrder kanan.

void postOrder(Simpul P)

{ if(P != null)

{ postOrder(P.kiri);

postOrder(P.kanan);

System.out.print(P.data + " \_ ");

}

}

}

//yaitu pada method postOrder,jika p=null maka posOrdernya kanan.

class ApliPohon

{

public static void main(String[] args)

{

Pohon P1 = new Pohon();

P1.akar = P1.tambah(P1.akar, 50);

P1.akar = P1.tambah(P1.akar, 60);

P1.akar = P1.tambah(P1.akar, 30);

P1.akar = P1.tambah(P1.akar, 40);

P1.akar = P1.tambah(P1.akar, 70);

P1.akar = P1.tambah(P1.akar, 55);

P1.akar = P1.tambah(P1.akar, 53);

P1.akar = P1.tambah(P1.akar, 57);

P1.akar = P1.tambah(P1.akar, 51);

P1.akar = P1.tambah(P1.akar, 54);

P1.akar = P1.tambah(P1.akar, 20);

P1.akar = P1.tambah(P1.akar, 10);

P1.akar = P1.tambah(P1.akar, 90);

//yaitu menjelaskan bahwa class Aplipohon maka pada method pohon maka isi akar 50,60,30,40,70,55,53,57,51,54,20,10,90.

System.out.print("Cetak pre order :");

P1.preOrder(P1.akar);

System.out.println("");

//yaitu untuk mencetak pada bagian preorder

System.out.print("Cetak in order :");

P1.inOrder(P1.akar);

System.out.println("");

//yaitu untuk mencetak bagian Cetak in order

System.out.print("Cetak post order :");

P1.postOrder(P1.akar);

System.out.println("");

}

}

//yaitu untuk output cetak post order.

KESIMPULAN

Kita dapat menyimpulkan bahwa pada pohon adalah sekumpulan akar cabang dan simpul yang terhubung secara heraki, maka pada pohn biner memiliki simpulnya hanya maksimal 2 anak dari cabang kanan dan kiri.