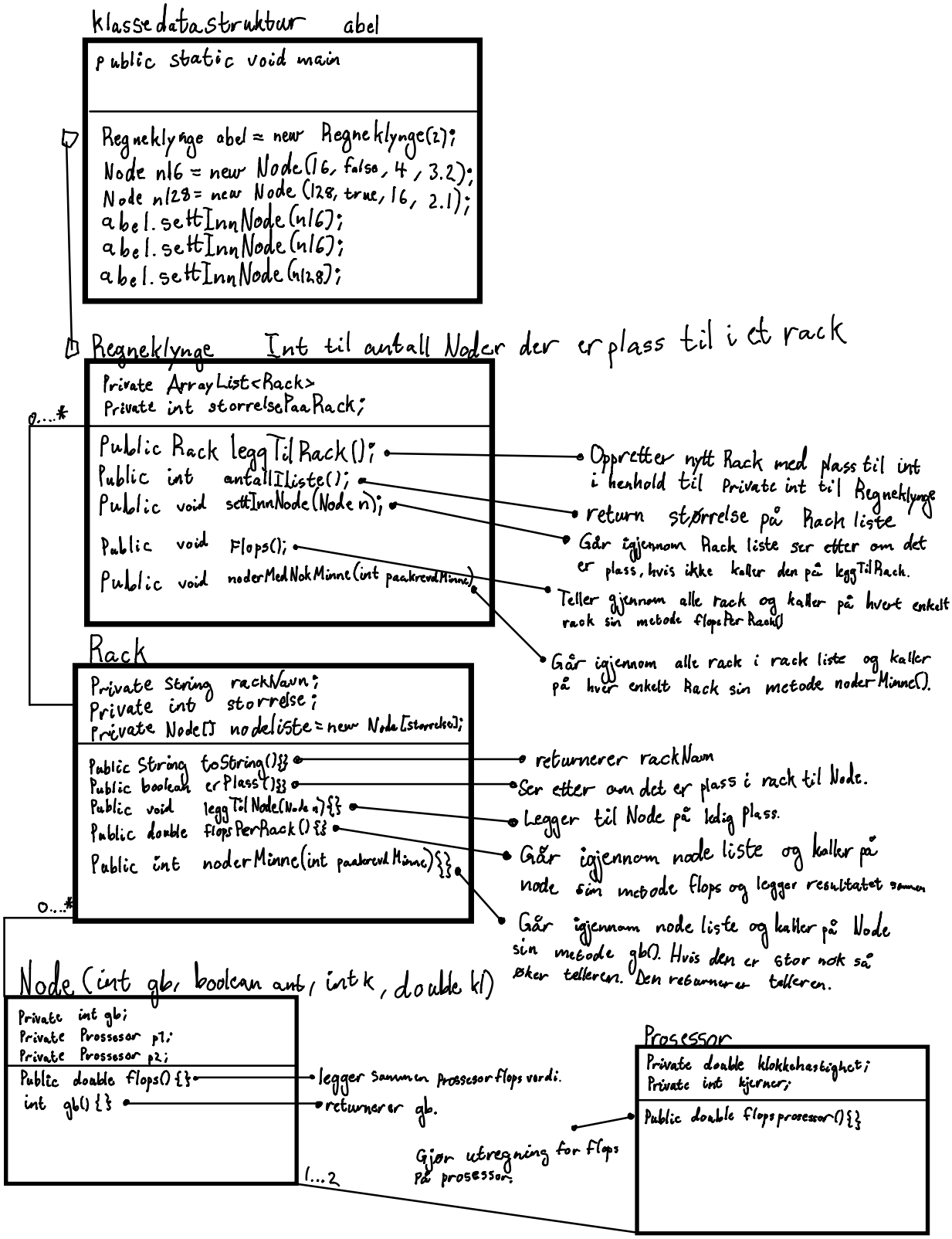
klassedatastruktur abel
public static void main
Regneklynge abel = new Regneklynge(2);
Node n16 = new Node(16, false, 4, 3.2);
Node n128 = new Node(128, true, 16, 2.1);
abel.settInnNode(n16);
abel.settInnNode(n16);
abel.settInnNode(n128);
Regneklynge
Int til antall Noder der er plass til i et rack
Private ArrayList<Rack>
Private int storrelsePaaRack;
Public Rack leggTilRack();
Public int antallIListe();
Public void settInnNode(Node n);
Public void flops();
Public void noderMedNokMinne(int paakrevdMinne);
Oppretter nytt Rack med plass til int i henhold til Private int til Regneklynge
return størrelse på Rack liste
Går igjennom Rack liste ser etter om det er plass, hvis ikke kaller den på leggTilRack.
Teller gjennom alle rack og kaller på hvert enkelt rack sin metode flopsPerRack()
Går igjennom alle rack i rack liste og kaller på hver enkelt Rack sin metode noderMinne().
Rack
Private String rackNavn;
Private int storrelse;
Private Node[] nodeliste = new Node[storrelse];
Public String toString(){}
Public boolean erPlass(){}
Public void leggTilNode(Node n){}
Public double flopsPerRack(){}
Public int noderMinne(int paakrevdMinne){}
returnerer rackNavn
Ser etter om det er plass i rack til Node.
Legger til Node på ledig plass.
Går igjennom node liste og kaller på node sin metode flops og legger resultatet sammen
Går igjennom node liste og kaller på Node sin metode gb(). Hvis den er stor nok så øker telleren. Den returnerer telleren.
Node(int gb, boolean ant, int k, double kl)
Private int gb;
Private Prossesor p1;
Private Prossesor p2;
Public double flops(){}
int gb(){}
legger sammen prossesor flops verdi.
returnerer gb.
Prosessor
Private double klokkehastighet;
Private int kjerner;
Public double flopsprosessor(){}
Gjør utregning for flops på prosessor.
0...*
0...*
1...2