IN1000 Obligatorisk oppgave 8—Datasenter

Sist endret 24. mars 2022.

Innledning

I denne oppgaven skal du lage et program for å holde oversikt over en rekke *regneklynger* (eng: computer cluster) og alle komponentene i en regneklynge. En slik regneklynge kan brukes til å fordele tunge beregninger på mange maskiner slik at de kan jobbe i parallell. På den måten kan en simulering som ville tatt en måned å kjøre på en vanlig maskin, kjøres på regneklyngen på noen timer i stedet. Det finnes flere regneklynger på UiO, hvorav Abel er den største.

Datasenter og regneklyngas bestanddeler

I denne oppgaven består et datasenter av en eller flere regneklynger. En regneklynge består av ett eller flere rack (et kabinett med skinner) hvor mange noder kan monteres over hverandre. En node er en selvstendig maskin med et hovedkort med én eller flere prosessorer og minne, i tillegg til en del andre ting. I denne oppgaven skal vi bare se på antall prosessorer (maks 2) og størrelsen på minnet. Du kan derfor anta at en node har en eller to prosessorer og et heltallig antall GB med minne.

Programdesign

Programmet skal designes med fire klasser som representerer henholdsvis noder, racks, regneklynge og datasenter. Et objekt av klassen Datasenter skal kunne referere til ett eller flere objekter av klassen Regneklynge. Et objekt av klassen Regneklynge skal kunne referere til ett eller flere Rack-objekter, der hvert Rack-objekt igjen refererer til en eller flere Node-objekter. Noen flere krav og tips til design av den enkelte klassen finner du videre i oppgaven. **Hver klasse og testprogrammet (hovedprogrammet) skal leveres i separate filer.**

Under overskriften **Oppgaver** nedenfor finner du beskrivelsen av hva programmet ditt skal kunne utføre. Dersom du ønsker mer hjelp på veien kan du ta utgangspunkt i det *offentlige grensesnittet* for hver klasse, som du finner i en egen fil sammen med denne oppgaven. Her finner du en dokumentert signatur (metode-hode) for de metodene klassene skal tilby utad, og som du skal implementere (skrive ferdig). Det kan i tillegg være nyttig å implementere hjelpemetoder (non-public metoder) som kun brukes innenfor den enkelte klasse (angis i Python med tegnet '_' foran metodenavnet).

Dersom du ønsker å arbeide mer selvstendig med oppgaven og lage dine egne grensesnitt for klassene er det flott – så lenge programmet omfatter disse fire klassene og har funksjonalitet som beskrevet under **Oppgaver** nedenfor. Kommenter i så fall dette øverst i hver klasse som ikke følger vedlagte forslag.

Klassen Node

Klassen Node skal kunne initiere nye objekter med ønsket minnestørrelse og antall prosessorer, og for øvrig tilby tjenester (metoder) som trengs i andre deler av programmet.

Klassen Rack

Klassen Rack skal lagre Node-objektene som hører til et rack i en liste. Vi skal kunne legge til noder i racket hvis det er færre enn maks antall noder der fra før. For enkelhets skyld skal vi anta at hvert rack i en regneklynge har plass til like mange noder (men dette maks-tallet kan variere fra regneklynge til regneklynge). Legg til andre instansvariabler og metoder etter behov.

Klassen Regneklynge

Klassen Regneklynge skal holde rede på en liste med racks, og må tilby en metode som tar et nodeobjekt som parameter og plasserer det i et rack med ledig plass. Hvis alle rackene er fulle, skal det lages et nytt Rack-objekt som legges inn i listen, og noden plasseres i det nye racket.
***Tips**:* Det kan være lurt å ta inn antall noder per rack i konstruktøren til Regneklynge.

Klassen Datasenter

Klassen Datasenter skal ta vare på en ordbok med regneklynger. Klassen skal også kunne lese inn regneklynger fra fil. For å få til dette trenger Datasenter en metode som tar imot et filnavn og oppretter en regneklynge basert på informasjonen i filen. Filnavnet vil bestå av regneklyngens navn og filendelsen .txt. Du kan bruke regneklyngens navn som nøkkel i ordboken og du kan anta at ingen regneklynger heter det samme.
Filen det skal leses inn fra har følgende format:
Max noder per rack
AntallNoder MinnePerNode AntallProsessorerPerNode
AntallNoder MinnePerNode AntallProsessorerPerNode
…

Klassen Datasenter trenger også en metode for å skrive ut informasjon om én spesifikk regneklynge og en metode for å skrive ut informasjon om alle regneklyngene i datasenteret.

Oppgaver

Programmet skal leveres med en fil per klasse og hovedprogram.

1. Datastrukturtegning

Les hele oppgaven før du tegner denne. Variabler (inkludert strenger) tegnes som navngitte bokser med verdien inni. Objekter (inkludert lister og ordbøker) tegnes som firkanter (runde hjørner) med instansvariabler inni. Klassenavn kan skrives over objektet med «:» foran. Referanser tegnes som piler fra variabelen de ligger i til objektet de refererer til. Du kan tegne for hånd eller bruke for eksempel Powerpoint eller et tegneprogram (***Tips***: Draw.io)

Tegn datastrukturen slik den ville sett ut etter vi hadde satt inn et objekt av Regneklynge i et objekt av Datasenter. Max antall noder per rack for denne regneklyngen skal være 2. I regneklyngen er det satt inn følgende noder:

* Node 1: 16 GB minne, 1 prosessor
* Node 2: 16 GB minne, 1 prosessor
* Node 3: 128 GB minne, 2 prosessorer

Tegningen skal leveres som en .pdf.

2. Lag klassene

Programmer klassene beskrevet under Programdesign. Husk å teste at hver klasse med alle metoder fungerer før du går videre til neste klasse (disse testene trenger du ikke levere).

3. Antall prosessorer og minnekrav

Lag en metode *antProsessorer(self)* i Regneklynge som returnerer det totale antall prosessorer i regneklyngen.
Noen programmer trenger mye minne, typisk et gitt antall GB med minne på hver node vi bruker. Vi er derfor interessert i å vite hvor mange noder som har nok minne til at vi kan bruke dem. Lag en metode *noderMedNokMinne(self, paakrevdMinne)* i Regneklynge som returnerer antall noder med minst *paakrevdMinne* GB minne.
Utvid klassene Node og Rack slik at de støtter implementeringen av disse metodene.

4. Testing underveis

Når du har laget klassen Regneklynge kan du bruke følgende verdier for å teste at alt fungerer slik det skal.

Lag en regneklynge, *abel*, og la det være plass til 12 noder i hvert rack. Legg inn 650 noder med 64 GB minne og en prosessor hver. Legg også inn 16 noder med 1024 GB minne og to prosessorer.
Sjekk hvor mange noder som har minst 32 GB, 64 GB og 128 GB minne. Finn totalt antall prosessorer, og sjekk hvor mange rack som brukes. Skriv ut svarene i terminalen. Utskriften kan f.eks. se slik ut:

Noder med minst 32 GB: 666
Noder med minst 64 GB: 666
Noder med minst 128 GB: 16
Antall prosessorer: 682
Antall rack: 56

5. Hovedprogram

Lag et Datasenter som leser inn informasjon om regneklyngene saga og abel fra testfiler som beskrevet ovenfor. For hver av regneklyngene skal du deretter skrive ut navnet på regneklyngen, antall racks, antall prosessorer og hvor mange noder som har minst 32 GB, 64 GB og 128 GB minne. Utskriften kan f.eks. se slik ut:

Informasjon om regneklynga abel
Antall rack: 56
Antall prosessorer: 682
Noder med minst 32 GB: 666
Noder med minst 64 GB: 666
Noder med minst 128 GB: 16