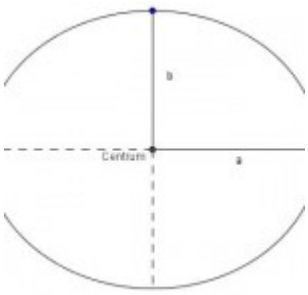
Praktisk information

Der mangler stadig gruppedannelser til de obligatoriske opgaver for 24 studerende!

Til en opgave i klassen:

En ellipses areal kan udregnes ud fra halvakserne *a* og *b* med følgende formel:

$$Areal = \pi \cdot a \cdot b$$



Omkreds af ellipse kan beregnes med formlen:

$$Omkreds = 2 \cdot \pi \cdot \sqrt{\frac{1}{2} \cdot (a^2 + b^2)}$$

Emner til lektion 4 (27/9)

Arv, Abstrakte klasser og Polymorphism.

Forberedelse til lektionen

Løs første del af den obligatoriske opgave.

Fra kapitel 8:

- Eksperimenter grundigt med eksemplerne i afsnit 8.1 og 8.2 (Listing 8.1 – 8.9). Det er vigtigt at blive fortrolige med begreberne *superclass/subclass*, *extends (arver fra), protected, super() (kald af constructor i superclass)* og *super.someMethod() (Kald af metode i superclass, som ellers er overskrevet).*
- Studer afsnit 8.3 og 8.4 grundigt.
- Læs afsnit 8.5 og tjek jeres forståelse med listen på side 402.

Løs PP8.1 side 406 og PP8.4 side 407.

Bearbejdelse af dagens emner

Vi starter med en diskussion af 1. del af den obligatoriske opgave, som i forhåbentlig er ved at få styr på.

Derefter gennemgår vi kap. 8. I kan forvente en ret grundig behandling af *constructorer* og deres opførsel i forbindelse med arv.

Undervejs tager vi et par opgaver i klassen (Shapes) og ser på jeres løsninger til *PP8.1* og *PP8.4*.

Så snakker vi lidt om næste uges emne, *Interface* og analyserer på arvehierarkiet til felterne i Matador. Se klassediagrammet på side 4 i beskrivelsen af den obligatoriske opgave.

Emner til lektion 5 (4/10)

Interfaces og Polymorphism

Forberedelse til næste lektion

Fra kap 9:

- Afsnit 9.1 er en ganske kort introduktion til begrebet *Polymorphism.* Læs det både før i kigger på det øvrige stof og bagefter. Giver det mere mening efter at de øvrige afsnit er behandlet?

- Eksperimenter med eksemplet i afsnit 9.2 (Listing 9.1 – 9.7). Det er et stort eksempel, så der skal bruges noget tid på det:
    - Studer hvordan klassen *Staff* indeholder et array af den abstrakte klasse *StaffMember*, som udfyldes af instancer af de konkrete klasser, der arver fra *StaffMember*, og hvordan den abstrakte metode *double pay()* implementers forskelligt i de konkrete klasser.
    - Bemærk brugen af *cast* i slutningen af constructoren i *Staff* , fx
      `((Hourly) staffList[3]).addHours(40);`.
      Hvorfor er det nødvendigt af *cast'*e til *Hourly*?
    - Prøv at forstå brugen af polymorphism i `void payDay()`.
- Afsnit 9.3 og 9.4 er en alt for kort beskrivelse af *Interfaces* og de viste eksempler (Listing 9.8 – 9.10) er efter min mening ikke særlig gode. Læs afsnittene igennem så vi kan bruge ret lang tid på at snakke om interfaces og programmere et par opgaver under lektionen.

Implementer videre på arve-hierarkiet i den obligatoriske opgave.