HØGSKOLEN I BERGEN
Avdeling for ingeniørutdanning

EKSAMEN I TOD062 – Grunnlegende programmering

KLASSE : 1Data (08HDATA) og
1 Informasjonsteknologi (08HINF)

DATO : 9. desember 2008

ANTALL OPPGAVER : 3
ANTALL SIDER : 6
VEDLEGG : 1 side

HJELPEMIDLER : Ingen. Kalkulator ikke lovlig

TID : 09.00 – 14.00 (5 klokketimer)

MÅLFORM : Bokmål

SENSOR(ER) : Ingen
FAGLÆRER(E) : Anya Helene Bagge
: Sven-Olai Høyland

MERKNADER : Ingen

Postboks 7030, 5020 Bergen. Tlf. 55 58 75 00, Fax 55 58 77 90
Besøksadr.: Nygårdsgt. 112, Bergen

Oppgåve 1 (20%)

a)

Vis nøyaktig hva som blir skrevet ut på skjermen når metoden **`oppgave1a(16,'*')`** nedenfor blir kjørt.

```
public static void oppgave1a(int m, char t){
      for (int i = 0; i < m; i= i+3){
            for (int j = 0; j < i/2; j++){
                  System.out.print(t);
            }
            System.out.println();
      }
}
```

b)

Vis nøyaktig hva som blir skrevet ut på skjermen når metoden **`oppgave1b(7)`** nedenfor blir kjørt.

```
public static void oppgave1b(int n){
      int s;
      for (int r = 0; r < n; r++){
            s = n - r;
            do{
                  System.out.print('#');
                  s--;

            } while (s > 0);
            System.out.println();
      }
}
```

c)

Skriv om metode **`oppgave1b(int n)`** slik at den bruker nøstete for-løkker for gi samme utskrift.

d)

Hva blir skrevet ut når metoden **`oppgave1d()`** blir kjørt. Gi ei kort begrunnelse i hvert tilfelle. Logiske verdier blir skrivet ut som true / false.

```
public static void oppgave1d(){
      String s1 = new String("Ein streng");
      String s2 = new String("Ein streng");

      System.out.println(10-4*2);
      System.out.println(3+7/2);
      System.out.println(5 % 2);
      System.out.println(s1.substring(1,5));
      System.out.println(s1 == s2);
      System.out.println(s1.equals(s2));
}
```

Oppgave 2 (50%)

Kaptein Jack 'Calico Jack' Rackham (mannen bak det svarte piratflagget) trenger hjelp til å holde orden på piratflåten sin. Du har fått i oppdrag å utvikle et datasystem for ham, og du er ganske sikker på at det er en dårlig idé å skuffe ham...

Feltvariabler skal være *private*. Du trenger ikke validere (sjekke) innleste data og parametre med mindre det er spesifisert i oppgaven. Du kan naturligvis benytte metodene du har laget når du lager nye metoder der det er hensiktsmessig.

a)

Definer en klasse `Skip` som representerer en skute med mannskap. Den skal ha følgende feltvariabler:

- `navn` – navn på skipet (streng)
- `kaptein` – kapteinens navn (streng)
- `kanoner` – antall kanoner (heltall)
- `mannskap` – antall pirater, inkludert kaptein og offiserer (heltall)
- `penger` – hvor mange *pieces of eight* skipet har plyndret (flyttall/desimaltall)

Definer følgende metoder for `Skip`:

- En konstruktør som tar navn, kaptein, kanoner og mannskap som parametre, og setter penger til 0.
- En konstruktør som tar navn og kapteinens navn som parametre, og setter kanoner til 20 og mannskap til 40. Du kan benytte konstruktørkjeding hvis du vil.
- `getPenger()`, `setPenger()` og `addPenger()`, som henter, setter og legger til penger. Du kan anta at det finnes hent/get- og sett-metoder for de andre feltvariablene.
- `toString()` -- instansmetode som returnerer navn, kaptein, kanoner, mannskap og penger som en streng, formattert slik: `"The Satisfaction (Capt. Morgan): 90 menn, 40 kanoner, 250000 pieces of eight"`

b)

Definer en klasse `Flåte` som representerer en samling med skip. Skipene skal lagres i en tabell. Klassen skal ha følgende feltvariabler:

- `skip` – tabell av `Skip`-objekter
- `antallSkip` – antall aktive skip i flåten (heltall)

Skriv følgende metoder til Flåte:

- En konstruktør som tar `maksAntall` som parameter og oppretter skipstabellen (uten aktive skip).
- `leggTilSkip` - instansmetode som tar et `Skip` som parameter og legger det til i tabellen. Metoden gjør ingenting dersom tabellen allerede er full.
- `finnSkip` – instansmetode som tar navnet på et Skip som parameter, søker gjennom skipene i tabellen, og returnerer skipet, eller null dersom det ikke ble funnet.

- `lesOppdatering` – instansmetode som går gjennom tabellen og ber brukeren skrive inn hvor mye hvert skip har plyndret for. Det innleste beløpet legges til pengene skipet allerede har.
- `skrivOversikt` – instansmetode som skriver ut oversikt over alle skipene, samt oppsummering av antall skip, antall menn og total mengde penger. Du kan evt. skrive en hjelpemetode i Skip-klassen, dersom det er hensiktsmessig.

```
Skip                Kaptein      Kanoner      Mannskap      Penger
-------------------------------------------------------------------
The Satisfaction    Morgan            40            90      250000
The William         Rackham           25            47      200000
The Royal Fortune   Roberts           30            77      125000
TOTAL                                 95           214      575000
```

c)

Anne Bonny har blitt valgt til kvartermester, og har ansvar for verdiene og fordeling av byttet. Det er ingen enkel jobb, og hun vil derfor gjerne ha utvidet datasystemet til å hjelpe med det også.

For hvert skip skal 25% av verdiene bli værende i skipskassen for å dekke investeringer og erstatninger for skader og omkomne. Resten fordeles til mannskapet etter følgende nøkkel:

- Kapteinen skal ha 3 andeler
- De øvrige offiserene skal ha 2 andeler hver
- Resten av mannskapet får 1 andel hver

Andelene rundes av nedover. Det som blir til overs, blir lagt til skipskassen.

Lag en instansmetode `fordeling()` i klassen `Skip` som oppdaterer skipskassen (feltvariablen `penger`) og skriver ut en fordelingsrapport for skipet. For eksempel:

```
Fordeling for 'The William' (mannskap 47):
  Rackham                          8181 =    8181
  Øvrige offiserer        6   *    5454 =   32724
  Øvrig mannskap         40   *    2727 =  109080
Totalt utdelt                             149985
Rest i kassen                              50015
```

Definer konstanter KAPTEINANDEL og OFFISERANDEL for å holde rede på fordelingsfaktorene.

Oppgåve 3 (30%)

a)

Skriv en metode

```
public static boolean erSortert(int tab[])
```

som undersøker om tabellen gitt som parameter er sortert stigende (de minste først). Om tabellen er sortert, skal metoden returnere **true**, elles **false**. Det kan være like elementer i tabellen.

b)

Skriv en metode som gitt en tabell av strenger, finn snittlengden av strengene. Gitt at tabellen inneholder de 4 strengene "abc", "123, "def", og "6174", så skal metoden returnere 3,25. Her må du bestemme første linjen av metoden selv.

c)

Skriv en metode som teller opp hvor mange elementer i en todimensjonal tabell som er større eller lik en nedre grense og mindre eller lik en øvre grense. Tabellen og grensene blir gitt som parmetre til metoden.

```
public static int antallMellomGrenser(int tab[][],
                                      int nedre, int ovre)
```

Eksempel: `t = [ 2, 7, 1]`
`[ 9, 5, 3]`
`[ 5, 1, 10]`

Kallet

```
antallMellomGrenser(t, 3, 8);
```

skal gi 4 som svar.

d)

Gitt følgende kode der du legger merke at klasse B arver fra klasse A.

```
public class A {
    ...
    public void f(){
      System.out.println("A sin f");
    }

    public void g(){
      System.out.println("A sin g");
    }
}
```

```
public class B extends A{
    ...
    public void f() {
      System.out.println("B sin f");
    }

    public void h(){
      System.out.println("B sin h");
    }

}
```

Oppgaven fortsetter på neste side.

En av setningene nedenfor er ulovlig og vil gi kompileringsfeil. Hvilken setning? Når vi ser bort frå den ulovlige setningen, hva blir skrivet ut?

```
A a = new A();
B b = new B();

a.f();
a.g();
a.h();

b.f();
b.g();
b.h();
```

Lykke til!

Vedlegg til eksamen i TOD062 desember 2008

Metodar i Scanner-klassen

Returverdi	Metode
int	nextInt ( )
double	nextDouble ( )
String	nextLine ( )

Klassen Utskrift

Du får kanskje bruk for følgjande static-metodar frå Utskrift-klassen:

repeterTeikn(char teikn, int antPos) - skriv *antPos* stk. av parameteren *teikn*

skrivHjHeiltal(int tal, int antPos) - skriv int-verdien *tal* høgrejustert på *antPos* posisjonar

skrivHjEinDesimal(double tal, int antPos) - skriv double-verdien *tal* med ein desimal på *antPos* posisjonar

skrivJustert(String tekst, int antPos, char justering) - Skriv *tekst* på *antPos* posisjonar venstrejustert, sentrert eller høgrejustert avhengig av *justering* ('v', 's', 'h')

Formatkodar for printf() og format():

"%-20s"	- String-data, 20 posisjonar, venstrejustert
"%6d"	- heiltal, 6 posisjonar, høgrejustert
"%04d"	- heiltal, 4 posisjoner, høyrejustert, fylt med ledende nuller
"%10.2f"	- flyttal, 10 posisjonar, 2 desimalar, høgrejustert

Klassen Random

Konstruktørar
Random ( ) - gir Random-objekt der ein brukar klokka for å gi startverdi
Random (int startverdi) - gir Random-objekt med gitt startverdi

Returverdi	Metode
double	nextDouble( ) - gir eit tilfeldig flyttal tal mellom 0 og 1
int	nextInt(n) - gir eit tilfeldig heiltal mellom 0 og n-1

Eit utvalg av metodar frå klassen String

Returverdi	Metode
int	**length** ( ) - gir antal teikn i ein streng.
char	**charAt** (int pos) - gir teikn i posisjon **pos** (start på 0).
boolean	**equals** (String b) - a.equals(b) sann om strengane a og b er like, usann elles.
boolean	**equalsIgnoreCase** (String b) - a.equalsIgnoreCase(b) sann om strengane a og b er like uavhengig av små og store bokstavar, usann elles.
int	**compareTo** (String s) - a.compareTo(b) gir eit negativ tal om a kjem først, 0 om strengane er like, og eit positiv tal om b kjem først (alfabetisk / ordboksordning).
int	**indexOf** (String s) - a.indexOf(b) gir -1 om s ikkje finst som delstreng i a, elles gir den første posisjon der b finst i a.
String	**substring** (int start, int slutt) - gir delstrengen frå og med posisjon *start* til (ikkje med) posisjon *slutt*.
String	**toLowerCase** ( ) - gir same streng, men alle bokstavane er små. (Viss ikkje anna er sagt kan de anta at det også gjeld for dei spesielle norske bokstavane).
String	**toUpperCase** ( ) - gir same streng, men alle bokstavane er store. (Viss ikkje anna er sagt kan de anta at det også gjeld for dei spesielle norske bokstavane).
String	**format**(String format, Object... args) – formaterer listen av argumenter i henhold til format-strengen. Returnerer resultatet. Static-metode.

Et utvalg av metodar fra klassen Integer

Returverdi	Metode
int	**parseInt** (String s) – gir tallverdien av s hvis s er et heltall. Kaster NumberFormatException hvis s ikke lar seg parse. Static-metode.