Prøveeksamen 100 %
PGR100 – Programmering 1

Oppgave 1

a) Studer følgende kodelinjer:

```
int first = 8;
int second = 19;
first = first + second;
second = first - second;
first = first - second;
```

Hvilken "effekt" har denne koden på variablene `first` og `second`?

b) Vis hva output blir når følgende kode blir utført:

```
for (int i = 1; i <= 5; i++) {
   for (int j = 1; j <= 6; j++) {
      System.out.printf ("%4d", (i * j));
   }
   System.out.println();
}
```

c) Studer koden i Vedlegg 1 og 2 og svar på følgende spørsmål:
I hvilke linjer (hvilken linje)

1) deklareres metodene
2) instansieres objekt
3) deklareres referanser
4) deklareres konstruktøren
5) deklareres metoden som returnerer opplysning om et objekt
6) kalles klassens metode(r)
7) deklareres set-metodene
8) blir objekt refererert av referanse
9) deklareres fields
10) kalles konstruktøren
11) gis klassens fields verdi

Oppgave 2

a) Skriv en metode `forekomst` som har en heltallsarray og et heltall som parametere. Metoden skal telle opp hvor mange ganger heltallet forekommer i arrayen og returnere dette antallet.

Eksempel: Hvis array'en inneholder tallene:
2, 1, 4, 3, 4, 3, 2, 4, 3, 5, 2, 5, 3, 1, 7
og tallet er 4, skal metoden returnere 3 (det er 3 4'ere i arrayen).

b) Skriv en metode `count` som har to parametere:

en `ArrayList` med `String`s (strenger)
et heltall

Metoden skal telle opp hvor mange av strengene i listen som inneholder like mange tegn som verdien av heltallsparameteren. Dette antallet skal returneres.

Hvis heltallsparameteren har verdien 4, og listen inneholder strengene
`"this" "is" "lots" "of" "fun" "for" "every" "Java" "programmer"`
skal metoden returnere 3.
Det er 3 strenger (`"this" "lots" "Java"`) som inneholder 4 tegn (tegnet " er ikke en del av strengene).

Oppgave 3

a) Skriv en klasse `PhoneCard` som representerer et forhåndsbetalt kontantkort som holder rede på beløpet som en bruker har å ringe for.

Klassen skal ha attributter (fields) for

beløpet det kan ringes for (`credit` – et desimaltall)
prisen per ringeminutt (`charge` – et desimaltall)
en unik identifikasjon (`id` – en `String`)
antall minutter det er ringt for totalt (`minutes` – et desimaltall)

Klassen skal bl.a. ha

to konstruktører
standard tilgangsmetoder
en `toString`-metode (retur: en `String` som viser objektets tisltand)

Den ene konstruktøren skal ikke ha parametere. Den andre konstruktøren skal ha parametere for de tre første attributtene – antall minutter det er ringt for totalt skal settes til 0.

Klassen skal også ha metoden

`call` med en parameter for et antall ringeminutter.

Dette antallet skal metoden gange med prisen per ringeminutt (`charge`), og det resulterende beløpet skal så trekkes fra `credit`. Hvis `credit` ikke er stor nok til å dekke det ønskede antallet ringeminutter, skal metoden beregne hvor mange minutter `credit` dekker, og endre berørte attributter (fields) ut fra dette. Metoden skal returnere forbruket (i kroner) for denne samtalen.

b) Skriv en klasse `PhoneCardProgram` med en `main`-metode som oppetter et `PhoneCard`–objekt ved bruk av konstruktøren med parametere. Programmet skal bruke klassens metoder på passende måte, slik at det kan lage output til Terminal Window som viser hva som skjer – som vist under.

```
Constructs a phone card (credit = 200 kr, minute charge = 0.5 kr, id = EasyPhoning 1234)

Makes a 10 minutes call
Used kr. 5.0
Credit left: 195.0
Minutes used (total): 10.0

Makes a 5 minutes call
Used kr. 2.5
Credit left: 192.5
Minutes used (total): 15.0

Makes a 1000 minutes call
Used kr. 192.5
Credit left: 0.0
Minutes used (total): 400.0

Card info:
Credit left: 0.0
Minute charge: 0.5
Minutes used: 400.0
Id: EasyPhoning 1234
```

Legg merke til at `Credit left` ble 0 da det ble (forsøkt) gjort en samtale på 1000 minutter. Antall minutter ble beregnet ut fra hva som var igjen av `credit` på dette tidspunktet.

Vedlegg 1

```
1  public class GroceryItem {
2      private String name;
3      private double pricePerUnit;
4
5      public GroceryItem() {
6          this(null, -1.0);
7      }
8
9      public GroceryItem(String name, double pricePerUnit) {
10         setName(name);
11         setPricePerUnit(pricePerUnit);
12     }
13
14     public void setName(String name) {
15         this.name = name;
16     }
17
18     public void setPricePerUnit(double pricePerUnit) {
19         this.pricePerUnit = pricePerUnit;
20     }
21
22     public String getName() {
23         return name;
24     }
25
26     public double getPricePerUnit() {
27         return pricePerUnit;
28     }
29
30     public String getData() {
31         return getName() + " " +
32         getPricePerUnit();
33     }
34 }
```

Vedlegg 2

```
51          GroceryItem item1;
52          item1 = new GroceryItem("Lettmelk", 8.55);
53          GroceryItem item2;
54
55          String name = item1.getName();
56
57          System.out.println(item1.getPricePerUnit());
```