Universitetet i Bergen
Det matematisk–naturvitenskapelige fakultet
Institutt for informatikk

BOKMÅL

EKSAMEN I EMNET INF100 Grunnkurs i programmering (Programmering 1) - Løsningsforslag

Tillatte hjelpemiddel: Alle trykte og skrevne hjelpemiddel er tillatt.

Oppgave 1 (20 %)

a) (4%) Hvilket tall vises på skjermen når `Klient1a` kjøres? 2

b) (4%) Hvilket tall vises på skjermen når `Klient1b` kjøres? 0

c) (4%) Hvilket tall vises på skjermen når `Klient1c` kjøres? 4

d) (4%) Hvilket tall vises på skjermen når `Klient1d` kjøres? 2

e) (4%) Hvilket tall vises på skjermen når `Klient1e` kjøres? 8

Oppgave 2 (38%)

a) (15%)

```
/** Løsningsforslag oppgave 2a */

public class Oppg2a {

  public static void main( String[] args ) {

    // Oppretter en terning å spille med
    Terning terning = new Terning();
    int poeng = 0;  // Teller poengene
    System.out.print("Du har " + poeng + " poeng. Flere omganger(j/n)? ");
    char svar = Terminal.lesString().charAt(0);
    boolean fortsett = svar=='j' || svar=='J';

    // Gjenta så lenge bruker har både lyst og lov
    while (fortsett) {
      int forrige = terning.hentVerdi();
      int verdi = terning.kast();
      System.out.println("Poeng i omgangen: " + verdi);

      // Sjekker om vi må avbryte spillet
      if (verdi < forrige) {
        // Avbryt
        poeng = 0;
        fortsett = false;
      }
      else {
        // Ny omgang
        poeng += verdi;
        System.out.print("Du har " + poeng + " poeng. Flere omganger(j/n)? ");
        svar = Terminal.lesString().charAt(0);
        fortsett = svar=='j' || svar=='J';
      }
    }
    System.out.println("Spillet avsluttes med " + poeng + " poeng.");
  }
}
```

b) (15%)

```
/** Klasse som representerer flere terninger.
  * Når vi kaster terningene får vi en tilfeldig verdi
  * mellom antall terninger og ANTALL_SIDER*(antall terninger) */
public class FlereTerninger {

  /** Verdien vi fikk sist vi kastet terningen */
  private Terning[] terninger;

  /** Oppretter tabellen med terninger. */
  public FlereTerninger(int antall) {
    if (antall > 0) {
      terninger = new Terning[antall];
      for (int i=0; i<terninger.length; ++i)
        terninger[i] = new Terning();
    }
  }

  /** Kaster terningene på nytt, og returnerer totalverdien vi får */
  public int kast() {
    int verdi = 0;
    if (terninger != null)
      for (int i=0; i<terninger.length; ++i)
        verdi += terninger[i].kast();
    return verdi;
  }

  /** Returnerer verdien vi fikk sist vi kastet terningen med indeks i */
  public int hentVerdi(int i) {
    if (i >= 0 && i<terninger.length)
      return terninger[i].hentVerdi();
    else
      return 0;
  }

  /** Returnerer total verdi vi fikk sist vi kastet terningene */
  public int hentVerdi() {
    int verdi = 0;
    if (terninger != null)
      for (int i=0; i<terninger.length; ++i)
        verdi += terninger[i].hentVerdi();
```

```
        return verdi;
      }

    }
```

c) (8%)

```
/** Løsningsforslag oppgave 2c */

public class Oppg2c {

  public static void main( String[] args ) {

    // Oppretter terninger å spille med
    System.out.print("Antall terninger du vil spille med: ");
    int antall = Terminal.lesInt();
    FlereTerninger terninger = new FlereTerninger(antall);

    int poeng = 0;  // Teller poengene
    System.out.print("Du har " + poeng + " poeng. Flere omganger(j/n)? ");
    char svar = Terminal.lesString().charAt(0);
    boolean fortsett = svar=='j' || svar=='J';

    // Gjenta så lenge bruker har både lyst og lov
    while (fortsett) {
      int forrige = terninger.hentVerdi();
      int verdi = terninger.kast();
      System.out.println("Poeng i omgangen: " + verdi);

      // Sjekker om vi må avbryte spillet
      if (verdi < forrige) {
        // Avbryt
        poeng = 0;
        fortsett = false;
      }
      else {
        // Ny omgang
        poeng += verdi;
        System.out.print("Du har " + poeng + " poeng. Flere omganger(j/n)? ");
        svar = Terminal.lesString().charAt(0);
        fortsett = svar=='j' || svar=='J';
      }
    }
    System.out.println("Spillet avsluttes med " + poeng + " poeng.");
  }
}
```

Oppgave 3 (42 %)

//

```
/** Klasse som representerer en kommune. */

import java.io.*;

public class Kommune {

  /** Feltskilletegn for tkstfiler: | */
  public static final char FELTSKILLE = '|';

  // Oppgave 3a
  /** Kommunenavn */
  private String navn;

  /** Antall innbyggere i fast arbeid */
  private int sysselsatte;

  /** Antall innbyggere som søker fast arbeid */
  private int ledige;

  /** Antall andre innbyggere */
  private int andre;

  // Oppgave 3b
  /** Oppretter en kommune. */
  public Kommune(String n, int s, int l, int a) {
    navn = n;
    settSysselsatte(s);
    settLedige(l);
    settAndre(a);
  }

  // Oppgave 3c
  /** Henter ut kommunenavnet */
  public String hentNavn() {
    return navn;
  }
```

```
// Oppgave 3c
/** Henter ut antall innbyggere i fast arbeid */
public int hentSysselsatte() {
  return sysselsatte;
}

// Oppgave 3c
/** Henter ut antall innbyggere som søker fast arbeid */
public int hentLedige() {
  return ledige;
}

// Oppgave 3c
/** Henter ut antall andre innbyggere */
public int hentAndre() {
  return andre;
}

// Oppgave 3d
/** Henter ut totalt antall innbyggere */
public int hentAntallInnbyggere() {
  return sysselsatte + ledige + andre;
}

// Oppgave 3d
/** Henter ut arbeidsledigheten */
public double hentLedighet() {
  return (100.0*ledige)/(sysselsatte + ledige);
}

// Oppgave 3e
/** Setter ny verdi til antall sysselsatte */
public void settSysselsatte(int s) {
  if (s >= 0)
    sysselsatte = s;
}

// Oppgave 3e
/** Setter ny verdi til antall arbeidssøkende */
public void settLedige(int l) {
```

```
    if (l >= 0)
      ledige = l;
  }

  // Oppgave 3e
  /** Setter ny verdi til antall andre innbyggeree */
  public void settAndre(int a) {
    if (a >= 0)
      andre = a;
  }

  // Oppgave 3f
  /** Skriver alle data til fil */
  public void skriv(PrintWriter strøm) {
    strøm.println(navn + FELTSKILLE + sysselsatte + FELTSKILLE
                  + ledige + FELTSKILLE + andre);
  }

  // Oppgave 3g
  /** Leser alle data fra fil */
  public void les(BufferedReader strøm) throws IOException {
    String linje = strøm.readLine();
    int skille1 = linje.indexOf(FELTSKILLE);
    int skille2 = linje.indexOf(FELTSKILLE,skille1+1);
    int skille3 = linje.indexOf(FELTSKILLE,skille2+1);
    navn = linje.substring(0,skille1);
    settSysselsatte(Integer.parseInt(linje.substring(skille1+1,skille2)));
    settLedige(Integer.parseInt(linje.substring(skille2+1,skille3)));
    settAndre(Integer.parseInt(linje.substring(skille3+1)));
  }

}

//
```

//

```
/** Klasse som representerer et fylke. */

import java.io.*;

public class Fylke {

  // Oppgave 3h
  /** Fylkets navn */
  private String navn;

  /** Kommunene som fylket består av */
  Kommune[] kommuner;

  /** Antall kommuner som fylket består av */
  int antall;


  // Oppgave 3i
  /** Oppretter et fylke. */
  public Fylke(String n, int maxAntall) {
    navn = n;
    if (maxAntall > 0)
      kommuner = new Kommune[maxAntall];
    antall = 0;
  }

  // Oppgave 3j
  /** Henter ut kommunen med indeks i */
  public Kommune hentKommune(int i) {
    if (i>=0 && i<antall)
      return kommuner[i];
    else
      return null;
  }

  // Oppgave 3k
  /** Legger til en ny kommunen med oppgitte data */
  public Kommune nyKommune(String navn, int sysselsatte, int ledige, int andre) {
```

```
    if (antall >= kommuner.length)
      return null;
    else {
      kommuner[antall++] = new Kommune(navn, sysselsatte, ledige, andre);
      return kommuner[antall-1];
    }
  }

  // Oppgave 3l
  /** Fjerner kommunen med indeks i */
  public Kommune fjernKommune(int i) {
    if (i>=0 && i<antall) {
      Kommune k = kommuner[i];
      for (int j=i; j<antall-1; ++j)
        kommuner[j] = kommuner[j+1];
      antall--;
      return k;
    }
    else
      return null;
  }

  // Oppgave 3m
  /** Skriver alle kommunene til fil */
  public void skriv(String filnavn) throws IOException {
    FileWriter filskriver = new FileWriter(filnavn);
    PrintWriter strøm = new PrintWriter(filskriver);
    strøm.println(antall);
    for (int i=0; i<antall; ++i)
      kommuner[i].skriv(strøm);
  }

  // Oppgave 3n
  /** Slår sammen de to minste kommunene */
  public Kommune slåSammen() {
    int minste = -1;
    int nestMinste = -1;
    // Gå gjennom kommunene, og finn de to minste
    for (int i=0; i<antall; ++i)
      if (minste < 0)  // Ingen er minst så langt, denne må være det (i==0)
        minste = i;
```

```
      else if (kommuner[i].hentAntallInnbyggere()
               < kommuner[minste].hentAntallInnbyggere()) {
        // Minste så langt, oppdater både minste og nest minste
        nestMinste = minste;
        minste = i;
      }
      else if (nestMinste < 0)
        // Ingen er nest minst så langt, denne må være det (i==1)
        nestMinste = i;
      else if (kommuner[i].hentAntallInnbyggere()
               < kommuner[nestMinste].hentAntallInnbyggere())
        // Nest minste så langt, oppdater nest minste
        nestMinste = i;

    if (minste >= 0 && nestMinste >= 0) {
      // Kommunene finnes. Finn data for ny kommune
      Kommune minsteKom = kommuner[minste];
      Kommune nestMinsteKom = kommuner[nestMinste];
      String knavn = nestMinsteKom.hentNavn() + "/" + minsteKom.hentNavn();
      int sysselsatte = nestMinsteKom.hentSysselsatte() + minsteKom.hentSysselsatte();
      int ledige = nestMinsteKom.hentLedige() + minsteKom.hentLedige();
      int andre = nestMinsteKom.hentAndre() + minsteKom.hentAndre();
      // Fjerner kommunene.
      fjernKommune(minste);
      if (minste < nestMinste)
        // Posisjonen til nest minste er forandret!
        fjernKommune(nestMinste-1);
      else
        fjernKommune(nestMinste);
      // Oppretter ny kommune
      return nyKommune(knavn, sysselsatte, ledige, andre);
    }
    else
      return null;
  }

}

//
```