Lösningsförslag till tentamen för TDA540 Objektorienterad Programmering

Institutionen för Datavetenskap
CTH HT-15, TDA540

Dag: 2016-01-16, *Tid:* 14.00-18.00

Uppgift 1

Metoden konverterar en decimal till en annat talsystem med bas `base`. I det här fallet til binärtalsystemet. Metoden är rekursiv. Utskriften är 42 i binär:

```
101010
```

Uppgift 2

Utskriften blir:

```
1, 2, 6
4, 5, 6
```

Uppgift 3

En lokal variabel är synlig inom det programblock där variabeln deklareras. Variabeln `b` är alltså okänd utanför `do`-blocket. En korrekt kod har följande utseende:

```
boolean b;
do {
  b = Math.random() < 0.5;
} while (b);
```

Uppgift 4

```
public static List<Integer> factorize(int n) {
  List<Integer> factors = new ArrayList<Integer>();
  final int BASE = 10;

  while (n > 0) {
    factors.add(0, n % BASE);
    n /= BASE;
  }

  return factors;
}

public static int sum(List<Integer> xs) {
  int res = 0;
  for (int x : xs)
    res += x;
  return res;
}

public static boolean isKeithNumber(int n) {
  List<Integer> numbers = factorize(n);
  int m = 0;

  do {
    m = sum(numbers);

    if (n == m)
      return true;
    else {
      numbers.remove(0);
      numbers.add(m);
    }
  } while (n > m);

  return false;
}
```

Uppgift 5

```
public class Keith {
  public static void main(String[] args) {
    cmdKeith();
  }

  // Kommandofönster version
  public static void cmdKeith() {
    boolean done = false;
    Scanner sc = new Scanner(System.in);

    while (!done) {
      System.out.print("Give a number: ");
      if (sc.hasNext()) {
        int n = sc.nextInt();
        if (n < 10)
          System.out.println("The number should be 10 or higher!");
        else {
          if (isKeithNumber(n))
            System.out.println(n + " is a Keith number!");
          else
            System.out.println(n + " is not a Keith number!");

        }
      } else done = true;
    }
  }

  // Dialogrutor version
  public static void dialogKeith() {
    boolean done = false;

    while (!done) {
      String input = JOptionPane.showInputDialog("Give a number: ");
      if (input != null) {
        Scanner sc = new Scanner(input);
        int n = sc.nextInt();
        if (n < 10)
          JOptionPane.showMessageDialog(null, "The number should be 10 or higher!");
        else {
          String msg = n + " is " + (isKeithNumber(n) ? "" : "not") + " a Keith number!";
          JOptionPane.showMessageDialog(null, msg);
        }
      } else done = true;
    }
  }
}
```

Uppgift 6

```
public static int[] getIndexOfMinValues(int[] xs) {
  int min   =  minimum(xs);
  int[] res = new int[frequency(xs, min)];
  int pos   = 0;

  for (int i = 0; i < xs.length; i++)
    if (xs[i] == min) res[pos++] = i;

  return res;
}

private static int minimum(int[] xs) {
  int min = xs[0];

  for (int x : xs)
    if (x < min) min = x;

  return min;
}
```

```
private static int frequency(int[] xs, int val) {
  int count = 0;

  for (int x : xs)
    if (x == val) count++;

  return count;
}

// More efficient alternative
public static int[] getIndexOfMinValuesEff(int[] xs) {
  int[] minima = new int[xs.length];
  int count    = 0;
  int min      = 0;

  if (xs.length > 0) {
    minima[count++] = 0;
    min             = xs[0];
  }

  for (int i = 1; i < xs.length; i++) {
    if (xs[i] < min) {
      minima[0] = i;
      min       = xs[i];
      count     = 1;
    } else if (xs[i] == min) {
      minima[count++] = i;
    }
  }

  return Arrays.copyOf(minima, count);
}
```

Uppgift 7

```
public static int[][][] blur(int[][][] samples, int n) {
  int[][][] newSamples = new int[samples.length][samples[0].length][3];
  double[][] filter = makeFilter(n);

  for (int row = 0; row < samples.length; row++)
     for (int col = 0; col < samples[row].length; col++)
        for (int c = 0; c < samples[row][col].length; c++)
           newSamples[row][col][c] = applyFilter(filter, samples, row, col, c);

  return newSamples;
}

public static int applyFilter(double[][] filter, int[][][] samples, int row, int col, int c) {
  int res = 0;
  int offX = filter.length / 2;
```

```java
  int offY = offX > 0 ? filter[0].length / 2 : 0;

  for (int i = 0; i < filter.length; i++)
     for (int j = 0; j < filter[i].length; j++)
        try {
           res += filter[i][j] * samples[row + i - offX][col + j - offY][c];
        } catch (IndexOutOfBoundsException e) {
           res += filter[i][j] * samples[row][col][c];
        }
  return res;
}

public static double[][] makeFilter(int n) {
  double[][] filter = new double[n][n];  // initialised with zeros

  for (int i = 0; i < n; i++)
    filter[i][i] = 1.0 / n;

  return filter;
}
```

Uppgift 8

```java
public static String generate(String s, int n) {
  Random rnd = new Random();
  String res = "";

  if (s.length() > 0) {
    for (int i = 0; i < n; i++)
      res += s.charAt(rnd.nextInt(s.length()));
  }

  return res;
}
```

Uppgift 9

```java
public static class ILanguage {
  private final String vowels = "aouåeiyäö";

  public String toI(String swe) {
    String i = "";

    for (char ch : swe.toCharArray())
      if (isVowel(ch))
        i += Character.isLowerCase(ch) ? "i" : "I";
      else
        i += ch;

    return i;
  }

  // Alternative
  public String toII(String s) {
    return s.replaceAll("[aouåeiyäö]", "i")
            .replaceAll("[AOUÅEIYÄÖ]", "I");
  }

  private boolean isVowel(char ch) {
    return vowels.indexOf(Character.toLowerCase(ch)) >= 0;
  }
}
```

Uppgift 10

```
private static void advice(int[] a, int[] b) {
  boolean success = false;
  double avgA = average(a);
  double avgB = average(b);

  for (int i = 0; i < a.length && !success; i++) {
    if (a[i] < avgA && a[i] > avgB) {
      System.out.println("Move " + a[i] + " from A to B!");
      success = true;
    }
  }

  for (int i = 0; i < b.length && !success; i++) {
    if (b[i] < avgB && b[i] > avgA) {
      System.out.println("Move " + b[i] + " from B to A!");
      success = true;
    }
  }

  if (!success)
    System.out.println("Impossible");
}

private static double average(int[] arr) {
  double sum = 0;

  for (int i : arr)
    sum += i;

  return sum / arr.length;
}
```

Uppgift 11

Exempel på olika tänkbara klasser och attribut (finns fler)

```java
public class Player {
  private final String name;
  private List<Card> cards;
  private int numberOfFours;

  public Player(String name, List<Card> cards) {
    this.name  = name;
    this.cards = cards;
    numberOfFours = 0;
  }
}

public enum Suit {
  HEARTS, DIAMONDS, CLUBS, SPADES
}

public enum Rank {
  ACE, KING, QUEEN, JACK, TEN, NINE, EIGHT, SEVEN, SIX, FIVE, FOUR, THREE, TWO
}

public class Card {
  private final Suit suit;
  private final Rank rank;

  public Card(Suit suit, Rank rank) {
    this.suit = suit;
    this.rank = rank;
  }
}

public class Deck {
  private List<Card> cards;

  public Deck(List<Card> cards) {
    this.cards = cards;
  }

  public Deck() {
    cards = new ArrayList<>();

    for (Suit suit : Suit.values())
      for (Rank rank : Rank.values())
        cards.add(new Card(suit, rank));
  }

  public List<Card> take(int n) {
    // Randomly take n cards from cards
    return new ArrayList<>();
  }
}

public class GoFish {
  private List<Player> players;
  private Deck ocean;

  public GoFish(List<Player> players, Deck ocean) {
    this.players = players;
    this.ocean   = ocean;
  }
}
```

Så här kan man konstruera modellen

```
private GoFish buildGame() {
  final int NRCARDS = 7;
  Deck ocean = new Deck();
  String[] names = {"Olle", "Fia", "Kalle"};
  List<Player> players = new ArrayList<>();

  for (String name : names) {
    List<Card> cards = ocean.take(NRCARDS);
    players.add(new Player(name, cards));
  }

  return new GoFish(players, ocean);
}
```