Lösningsförslag till tentamen för TDA540 Objektorienterad Programmering

Institutionen för Datavetenskap

CTH HT-17, TDA540

Dag: 2018-04-06, *Tid:* 14.00-18.00

Uppgift 1

a) `class` används för en klassdeklaration som är ett mall för att skapa objekt

`public` en modifierare som antyder att enheten (instansvariabel, metod, etc.) är tillgängligt för alla

`static` en modifierare som anger att en enhet (variabel eller metod) tillhör själva klassen och är gemensamt för alla klassens objekt

`void` returtyp som antyder att en metod har inget returvärde

`int` används för att deklarera en variabel som kan spara en heltal

`new` skapar en nytt objekt av given klass

`for` används för att skapa en räknareloop

`if` används för selektering mellan två satser beroende på en

`break` slutar exekvera nuvarande loop boolesk uttryck

b)

Klass	*Signatur*	*Returtyp*	*Statisk*
`Votes`	`main(String[] args)`	`void`	ja
`Votes`	`countVotes(int nOptions, int[] votes)`	`void`	ja
`PrintStream`	`println(int i)`	`void`	nej

c)
```
17 int[] votes = {1, 2, 1, 1, 3, 2, 1, 4, 3, 2};
18 countVotes(4, votes);
4 int[] count = new int[nOptions];
5 for (int k = 0; ; k++)
6 int vote = votes[k];
7 count[vote - 1]++;
8 if (k == votes.length - 1) // k är 0, men votes.length är 10
```

d) Utskriften:

```
Option 1 got 4 votes
Option 2 got 3 votes
Option 3 got 2 votes
Option 4 got 1 votes
```

For every integer between 1 and nOptions, method `countVotes` prints the number of elements of `votes` that equal that integer ("votes" for that option).

e)
```
public static void countVotes(int nOptions, int[] votes) {
   Integer[] count = new Integer[nOptions];
   for (int k = 0; k < nOptions; k++)  // Integer[] is initialized to null not 0
      count[k] = 0;
   for (int k = 0; k < votes.length; k++) {
      int vote = votes[k];
      count[vote - 1]++;
   }
   for (int k = 0; k < count.length; k++) {
      System.out.println("Option " + (k + 1) + " got " + count[k] + " votes");
   }
}
```

Uppgift 2

- Variable `k` is used but not declared.
- `k + 1` is an expression, which cannot be used as a statement; it should be an increment statement.
- Det saknas en `return`-sats.

Förbättrade versionen:

```
public static int max_correct(int[] a) {
  int m = a[0];
  int k = 0;
  while (k < a.length) {
    if (a[k] > m)
```

```
            m = a[k];
          k++;  // eller k = k + 1;
        }
        System.out.println(m);
        return m;
    }
```

Uppgift 3

a) Utskriften:

```
1
4
7
```

b) Det är klassen `Throwable`.

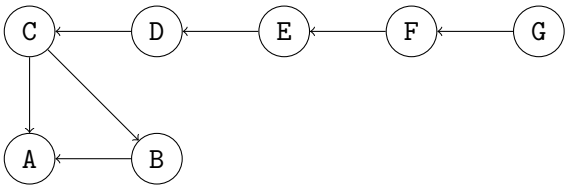
c) The program would not compile. Since `Exception` is a checked exception class, every statement that may throw it (that is, every call to `cc()`) has to be in a `try` block (or the exception be propagated). It doesn't matter that only the last call can actually throw an exception, nor that the actual type of thrown exception is `RuntimeException` which in unchecked: the compiler checks based on the declaration `throws Exception`.

Uppgift 4

```
class Data extends BasicData {

   public Data() {
      super();
   }

   public int argmin() {
      if (size() == 0)
         throw new RuntimeException("Empty dataset");
      int result = 0;
      for (int k = 0; k < size(); k++)
         if (get(k) < result)
            result = k;
      return result;
   }

   public double min() {
      return get(argmin());
   }

   public double sum() {
      double result = 0.0;
      for (int k = 0; k < size(); k++)
         result += get(k);
      return result;
   }

   public double mean() {
      return sum()/size();
   }

   public Data filter(double x) {
      Data result = new Data();
      for (int k = 0; k < size(); k++) {
         double d = get(k);
         if (d < x)
            result.put(d);
      }
      return result;
   }

   public Data centered() {
      Data result = new Data();
      double m = mean();
      for (int k = 0; k < size(); k++)
         result.put(get(k) - m);
      return result;
   }
}
```

Uppgift 5

a) Klassdiagrammet ser ut så här:



b) 1. 1
2. 0
3. 3
4. 0
5. 0
6. 0
7. 0
8. 0
9. 3
10. 3
11. 3

c) 1. Korrekt
2. Inkorrekt: ett interface kan inte instansieras
3. Inkorrekt: ett interface kan inte instansieras
4. Korrekt
5. Inkorrekt: man får en typfel för `C` är ingen subtyp av `G`
6. Korrekt
7. Korrekt
8. Inkorrekt: man får en typfeil för `ArrayList<C>` är ingen subtyp av `ArrayList<A>` (even if `C` är subtyp av `A`)