Object Oriented Programming
TDA547, DAT170

Krasimir Angelov, tel. 073-8228-179

2014-08-13

These are example questions similar but not equivalent to those that will be given on the actual exam.

1. Read the following program:

```
public static void process(int[] a) {
   a[0] = a[0] + a[1];
   a[1] = a[0] - a[1];
   a[0] = a[0] - a[1];
}
```

   - What will be the content of the array `a` after executing:

```
int[] a = new int[] {5,7};
process(a);
```

   - Describe the effect of the method on an arbitrary array.

2. In this task we do simple array processing:

   - A palindrome is a sequence of symbols that reads the same forward and reversed. For example "abba" and "aba" are palindromes but "abak" is not.

     Write a method:

     `public static boolean isPalindrome(char[] a)`

     which returns `true` if the sequence in the array `a` is a palindrome. (4p)

- Implement a class `Question2` which can be used to test whether a sequence is a palindrome or not. The class should have a main method which reads a line from the keyboard and then prints true or false depending on the input. For example:

  ```
  > java Question2
  abba
  true
  ```

  Here the user has entered "abba".

  Hint: Use a Scanner to read a line of text. After that, if s is the string then s.toCharArray() returns an array with all characters in the string.

3. Implement the method:

   ```
   public static List nub(List list)
   ```

   It takes as input a list of objects and creates from it another one which contains the same elements except that duplicated elements are filtered out.

4. Game of Life (mini). This is a one dimensional version of the game of life. We have a row with `N` number of cells. In every generation a cell is either alive or dead according to the rules:

   - if in the current generation the cell is alive but it has exactly two alive neighbours (i.e. to the left and to the right) then it dies in the next generation.
   - if the cell is dead in the current generation and it has no alive neighbours then it becomes alive in the next generation.
   - in all other cases the cell retains its state to the next generation.

   Implement a class `Game` with the following methods:

   - a constructor:

     ```
     public Game(int n)
     ```

     which creates the initial state of the game with `n` cells, where every cell has a randomly chosen state where there is an equal chance of being alive or dead.
   - a method:

     ```
     public void step()
     ```

which updates the current state of the game to a new generation by applying the rules of the game.

– a method:

`public void print()`

which prints the current state of the game on a single line, where an alive cell is shown as `*` and a dead one with a space.