Programming Paradigms
Written Exam (6 CPs)
19.09.2018

First name 


Last name 

Student number 

Signature 

Instructions for Students

• Write your name and student number on the exam sheet and on every solution sheet you hand in and also sign them.

• This is a closed book exam: the only resources allowed are blank paper and pens (do not use pencils).

• Write neatly and clearly. The clarity of your explanations will affect your grade.

• The duration of the exam is 2 hours.

Good luck!

Do not write in this space

<table>
<thead>
<tr>
<th>Exercise</th>
<th>Marks</th>
<th>Achieved</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>
Exercise 1 (20 marks)

a. (4 marks) Briefly describe the concept of abstract data types and the advantages they introduced with respect to imperative/procedural programming.

b. (4 marks) Briefly describe the concept of a symbol in Ruby and its difference to strings.

c. (4 marks) What is the key difference in a Prolog query between using variables or atoms as parameters (e.g., between `likes(gromit,cheese)` and `likes(X,Y)`)?

d. (4 marks) What is the following Haskell list comprehension producing?

\[
[(x,y) \mid x \leftarrow [1..3], \ y \leftarrow [2..4], \ x \neq y]
\]

e. (4 marks) When you move the execution of an Erlang program from a single-processor machine to a multi-core machine or a cluster of computers, do you have to rewrite or adapt your program? Explain your answer.

Exercise 2 (8 marks) Extend the Ruby class `Fixnum` with a method `square_root_times` that, if called for a number \(n\) and a code snippet, executes the code snippet \(\lceil \sqrt{n} \rceil\) times. For example, `5.square_root_times{ puts 'hello world!' }` produces

```
hello world!
hello world!
hello world!
```

You are not allowed to use a built-in Ruby function to compute the square root of numbers.

Exercise 3 (12 marks) Assume temperature data stored in an array `t`. Write a Ruby function `maxperiod(t,x)` that calculates the length of the longest (hot) period, where all temperature values are greater than \(x\). The result is printed to the console. For example, for \(t = [20, 25, 26, 23, 27]\) and \(x = 24\), the function prints The longest period greater than 24 is of length 2.

Exercise 4 (14 marks) Write a Prolog predicate `decode( L1, L2 )` that decodes a run-length encoded list `L1` into the uncompressed list `L2`. Assume a simple run-length encoding, where a sequence of \(N\) characters `X` is represented as a pair `(N,X)`; sequences of length 1 are also encoded. For instance, the call `decode( [[2,a), (1,b), (3,c)], L2 )` should instantiate `L2 = [a, a, b, c, c, c]`. 
Exercise 5 (10 marks) Write a Prolog program \texttt{pythagoras(A,B,C)} using the “generate and test” pattern, which for a given value \( C \) computes all possible values of \( A \) and \( B \) for which the theorem of Pythagoras holds, i.e., \( A^2 + B^2 = C^2 \) (\( A, B \) and \( C \) are integer numbers). For instance, \texttt{pythagoras(A,B,5)} returns \( A=3, B=4 \) and \( A=4, B=3 \), whereas \texttt{pythagoras(A,B,6)} fails.

Hint: You can use the predicate \texttt{between(L,U,X)} which generates all integers between \( L \) and \( U \), e.g., \texttt{between(0,3,X)} generates

\[
X = 0 ; \\
X = 1 ; \\
X = 2 ; \\
X = 3.
\]

Exercise 6 (8 marks) Write a Haskell function \texttt{rotate} that rotates a list by \( n \), \( n \geq 0 \), places to the left. Here are a few examples:

\[
\text{rotate [1,2,3,4,5] 0 } \Rightarrow [1,2,3,4,5] \\
\text{rotate [1,2,3,4,5] 2 } \Rightarrow [3,4,5,1,2] \\
\text{rotate [1,2,3,4,5] 7 } \Rightarrow [3,4,5,1,2]
\]

Exercise 7 (14 marks) Write a Haskell module that exports a function \texttt{split} that splits a list \( l \) at a given position \( n \) into two lists. The list and the split position are given as input parameter. The function returns a pair \((l_1,l_2)\) consisting of the two parts of the list; if \( n \leq 0 \), \( l_1 \) is empty and \( l_2 \) contains the input list; if \( n \) is greater than the length of the input list, \( l_1 \) contains the input list and \( l_2 \) is empty. Here are a few examples:

\[
\text{split [7,8,9,3,4] 2 } \Rightarrow ([7,8], [9,3,4]) \\
\text{split [7,8,9,3,4] -1 } \Rightarrow ([], [7,8,9,3,4]) \\
\text{split [7,8,9,3,4] 7 } \Rightarrow ([7,8,9,3,4], [])
\]

Exercise 8 (14 marks) Write an Erlang module that implements and exports a server to count positive and negative votes of a ballot. If the server receives the message “yes”, the counter of the positive votes is incremented by 1, whereas the message “no” increments the number of negative votes by 1. If the message “info” is received, the server sends a message containing the number of positive and negative votes to a process that is registered with an atom \texttt{observer}, and it displays the counters on the screen. The message “Bye” terminates the process. Show also the command to start the server from the command line.