Exercise 2 — assigned Thursday 15 February — due Thursday 29 February

New in this exercise: many new syntactic forms (the whole core lambda language except for recursive types) and big-step semantics. The primary reference for this exercise is Pierce, *Types and Programming Languages*, Chapter 11.

2.1 Extending the core lambda language

Language features you should implement: char type (ASCII), unit type, let bindings, general recursion (fix), record types, variant types. Update the concrete syntax description according to the examples given. Extend all previous code to handle the new syntactic forms.

2.2 Main program

Write a main program which will (1) read the program text from a file into a string, (2) invoke the parser to produce an abstract syntax tree for the program, (3) print the free variables of the program, (4) in case the program is closed (has no free variables), type-check the program, (5) in case the program has a type, evaluate it with respect to its call-by-value structural operational semantics (yielding a term), (6) and evaluate it according to big-step operational semantics (yielding a value).

2.3 Testing

Consider developing a testing framework.

2.4 Provided code

`exercise2.lhs`
`AbstractSyntax.lhs` (updated)
`cbntest1.corelambda`
`intFact.corelambda`
`sumn100.corelambda`
`takFastOpts.corelambda`
`takTest1Opts.corelambda`
`takTestOpts.corelambda`
`test11.corelambda`
`test12.corelambda`

```
test13.corelambda
test14.corelambda
test15.corelambda
test16.corelambda
test17.corelambda
test18.corelambda
test21.corelambda
test23.corelambda
test24.corelambda
test25.corelambda
test26.corelambda
test27.corelambda
test28.corelambda
test39.corelambda
test40.corelambda
```

2.5 What to submit

Prepare a project report (PDF, named `report.pdf`). If you want, you can prepare the code and the report together in literate Haskell (like this assignment).

Your project report should include your commentary on the code structure, interesting design decisions, etc. As an appendix, include the complete program listing as well as example executions.

In the project report, **please remember to describe your team composition and how the team collaborated on the task, and comment on the level of effort needed for the project**.

Use Canvas to submit a tar file with all the code and the report.

How to make tar files? You do so in the directory above the one you are packaging:

```
darko@m$ ls -lR tarexample/
total 0
drwxr-xr-x  3 darko  staff  96 Feb 16 16:44 code
-rw-r--r--  1 darko  staff   0 Feb 16 16:44 report.pdf
tarexample//code:
total 0
-rw-r--r--  1 darko  staff  0 Feb 16 16:44 Main.lhs
darko@m$ tar cf - tarexample >tarexample.tar
darko@m$ tar tf tarexample.tar
tarexample/
tarexample/code/
```

```
tarexample/report.pdf
tarexample/code/Main.lhs
```

2.6 Oral presentations

Teams will give oral presentations in class on Thursday 29 February. They can be informal, they can include code walkthroughs and live demonstrations. All team members should describe their contributions. If you have used GHC language extensions, be ready to teach us how they work. Plan on 15–20 minutes per team.