University of Amsterdam
System and Network Engineering Lab
Dr Clemens Grelck

Compilerbouw
2017/2018
March 21, 2018

Assignment Series 6

Code Generation

Assignment 21: Code Generation

Consider the following CiviC function definition.

```
int factorial( int x)
{
  int res;
  if (x <= 1) res = 1;
  else res = x * factorial( x - 1);
  return res;
}
```

a) Manually generate CiviC-VM assembly code for the above function definition making use of labels to mark destinations of jump instructions.

b) Point out the relationship between assembly code and source code through line comments in the assembly code.

c) Add the number of bytes required for each line of CiviC-VM assembly code. Assume here jump instructions would take byte code offsets as arguments and not labels.

d) Compute the proper byte code offset for each jump instruction; consult the CiviC-VM manual for details on individual instructions.

Assignment 22: Compilation Schemes Revisited

Devise a compilation scheme that replaces each occurrence of a `for`-loop in the body of a CiviC function by semantically equivalent CiviC code that makes use of a `while`-loop instead. As a simplification consider only `for`-loops without a step specification and assume that CiviC would support arbitrary interleaving of variable declarations and statements in function bodies following the example of C99.

Assignment due date: April 1, 2018