COMP 3610 Tutorial 3

17 August, 2023

Exercise 1

1. Without considering type-checking, write a program P in IMP extended with functions and function types (up to and including definitions in Section 6) that does not include while loops, but will run forever without getting stuck.
2. Show either that P is well-typed or explain why it cannot be.

Exercise 2

1. Write a program P in IMP extended with functions, function types, and recursive function definitions (up to and including definitions in Section 7) that, without using a while-loop or locations, returns a function which, given a non-negative integer, computes and returns the factorial of that integer.
2. Show that P is well-typed as follows:
$$\{\} \vdash P : int$$
3. Show the steps for running the program $P2$.

Exercise 3

The Ackermann Function is commonly defined on non-negative integers as follows:

$$\begin{array}{lcl} A(0,n) & = & n+1 \\ A(m+1,0) & = & A(m,1) \\ A(m+1,n+1) & = & A(m,A(m+1,n)) \end{array}$$

1. Write a program P in IMP extended with functions, function types, and recursive function definitions (up to and including definitions in Section 7) that encodes this function.
2. Can you write this program using while-loops instead of recursive functions? If yes, show the code, if no, explain why not.

HINT: Our language does not feature subtraction, but we do have negative integer constants.

Exercise 4

1. It turns out that in the variant of IMP with products, sums, and records (up to and including definitions in Section 8, excluding mutable stores), booleans and if-expressions are redundant. Show how to encode them.

2. It turns out that in the variant of IMP with functions and function types (up to and incuding the definitions in Section 6), booleans and if-expressions are redundant. Show how to encode them.

Exercise 5

1. Write a program P in the variant of IMP with mutable stores (up to and including definitions in Section 8) that, without using a while-loop or recursive let-expressions, returns a function which, given a non-negative integer, computes and returns the factorial of that integer.

2. Show that P is well-typed as follows:

$$\{\} \vdash P : int$$

3. Show the steps for running the program $P2$.