COSC 5010. Formalizing the JVM in ACL2. Spring 2004
Examination
Due Tuesday 30 March 2004

- Answer these questions individually. You may only use the book, the homework, the class notes, the class website, and the ACL2 documentation. You may ask for help only from the instructor.
- Email your solutions to me at `cowles@uwyo.edu`.
- The subject line should identify the problem set, i.e. Exam Question 1.
- The body of the message should be a Lisp readable file in simple ascii text format.
- The first few lines of the body should contain your name.
- The file `m2.lisp` on the class website contains the definition of the M2 Virtual Machine.

Exam Question 1.

1. Extend M2 by formalizing the `INSTANCEOF` instruction. Below is a paraphrase the relevant section of the JVM specification, limited to the concepts of M2. You may answer merely by showing the definition of `execute-INSTANCEOF` and any supporting functions you need.

   Format:
   `(INSTANCEOF class)`
   Operand Stack:
   `..., objectref => ..., result`
   Description:
   The `objectref`, which must be a reference, is popped from the operand stack. The `class` given in the instruction must be a class name (string). If `objectref` is not null and is an instance of the named `class`, the `INSTANCEOF` instruction pushes the result of 1. Otherwise, it pushes a result of 0.

   The following rule is used to determine whether an `objectref` that is not null is an instance of the `class`. If S is the class of the object referred to by `objectref`, then `objectref` is an instance of `class` if S is `class` or S is a subclass of `class`.

2. Extend M2 by formalizing the instruction `INVOKESTATIC`, described below. You may merely exhibit the definition of `execute-INVOKESTATIC`.

   Format:
   `(INVOKESTATIC class method n)`
   Operand Stack:
   `..., arg1,arg2,...,argn => ...`
   Description:
   The `class` argument of the instruction must be a class name (string) in the current class table and `method` must be a method name within that `class`, with `n` formals.

   The `n` argument values are popped from the operand stack. A new frame is created on the call-stack for the method being invoked. The `n` argument values are bound to the formals of the `method`, with `arg1` bound to the first formal, and so on. This binding becomes the local variables of the new frame. The new frame is then made current, and the pc is set to the opcode of the first instruction of the method to be invoked. Execution continues with the first instruction of the `method`.

3. Exhibit an M2 class declaration of a new class called `"ConsCell"` with two fields, `"car"` and `"cdr"`, and two static methods. The first method, named `"cons"` has two formals, allocates a ConsCell object with the car and cdr fields set to the two arguments and returns (a reference to) the object. The second method, named `"len"` has one formal, `x`, and returns the length of the cdr chain of the ConsCell `x`. If `x` is not a ConsCell, its length is 0.