CSE 6341, Programming Project 4
Due Friday, November 10, 11:59 pm (20 points)

The goal of this project is to implement the abstract interpretation described in class. Arithmetic expressions will be evaluated to one of the abstract values from set \{NegInt, ZeroInt, PosInt, AnyInt, NegFloat, ZeroFloat, PosFloat, AnyFloat\}. Boolean expressions will be completely ignored; we will not look inside them and will not try to evaluate/check them or any of their subexpressions (i.e., in this project we will **not** use the techniques from slides 26-33 in the lecture notes).

Modify your code from Project 3 to implement this abstracted semantics. As part of this abstract interpretation, report checking errors if (1) the second operand of division is ZeroInt, ZeroFloat, AnyInt, or AnyFloat; (2) an uninitialized variable is being used. That is, the implementation should follow the description in slides 1-23 and should use the “more conservative” version described there. Examples were provided in the slides to illustrate the desired behavior.

**Set up**
Copy your implementation from Project 3 into .../proj/p4 and do `make clean`

**Restriction**
As with Project 3, the following restriction will be imposed on all input programs: no two variables have the same name. This also applies to variables that are in different blocks. You **do not** have to check that this restriction is satisfied by the input program: just assume that it is and implement your interpreter under this assumption.

**Details**
The abstract semantics was described in class (slides 1-23). A few additional details:

1) Do **not** print the program. Do **not** print the abstract state. Only print (1) expression values at print statements, and (2) error messages for static checking errors.

2) At a print statement of the form `print <expr>;` evaluate abstractly the expression to one of the 8 abstract values described above. Then print that abstract value (e.g., the string NegFloat) via `System.out.println(…).` Such printing is useful for testing, debugging, and grading.

3) `readint` and `readfloat` expressions evaluate to AnyInt and AnyFloat, respectively. You do not need to look in UNIX `stdin` since we are creating a static analysis that represents all possible executions for all possible valid program inputs, not for any particular program input. In this static analysis, we do assume that the execution will always find correct values in the input stream. Thus, your Project 4 code will never exit with error code `EXIT_FAILED_STDIN_READ.`
4) Static error “division by zero” should exit with error code EXIT_DIV_BY_ZERO_ERROR. Static error “use of uninitialized variable” should exit with error EXIT_UNINITIALIZED_VAR_ERROR.

Testing
Write many test cases and test your checker with them. Submit at least 5 test cases with your submission. The test cases you submit will not affect your score for the project. Put them in the same location as the provided file t1 and name them t2, ...

Submission
After completing your project, do
cd p4
make clean
cd ..
tar -cvzf p4.tar.gz p4

Then submit p4.tar.gz in Carmen.

General rules (copied from the course syllabus)

1) Your submissions must be submitted electronically via Carmen by midnight on the due date. The projects must compile and run on stdlinux. Some students prefer to implement the projects on a different machine, and then port them to stdlinux. If you decide to use a different machine, it is entirely your responsibility to make the code compile and run correctly on stdlinux before the deadline. In the past many students have tried to port to stdlinux too close to the deadline, leading to last-minute problems and missed deadlines.

2) Projects should be done independently. General high-level discussion of projects with other students in the class is allowed, but you must do all design, programming, testing, and debugging independently. Projects that show excessive similarities will be taken as evidence of cheating and dealt with accordingly. Code plagiarism tools may be used to detect cheating. See more details in the Syllabus under “Academic integrity”.

3) The projects are due by 11:59 pm on the due day. No exceptions will be made to this deadline: if you submit at 12:00 am, your submission will be late. Please plan your time carefully and do not submit in the last minute. You can submit up to 24 hours after the deadline; if you do so, your project score will be reduced by 10%. If you submit more than 24 hours after the deadline, the submission will not be accepted, and you will receive zero points for this project.

4) Accommodations for sickness and other special circumstances will be made based on university guidelines. Please contact me ahead of time to arrange for such accommodations.