This project is the last one for the course. Its goal is to extend your Project 4 with abstract
evaluation of boolean expressions. The possible abstract values for such expressions are True,
False, and AnyBool. Modify your code from Project 4 to implement this semantics. As part of
this abstract interpretation, report when dead code is identified, as described below.

Set up
Copy your implementation from Project 4 into \.../proj/p5 and do make clean

Restriction
As with Projects 3 and 4, 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 24-31). 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 run-time errors.

2) Dead code checking for if-then and if-then-else statements: if the condition of an if-then-else
statement evaluates to True, the else-branch is guaranteed to be dead code. Similarly, if the
condition of an if-then-else or if-then evaluates to False, the then-branch is guaranteed to be
dead code. In those cases, exit with a dead code error. Introduce a new exit code
EXIT_DEAD_CODE with value 6 in Interpreter.java. Use this exit code whenever dead code is
detected as described above. Note that slide 26 shows something different: there, a warning is
printed and the execution continues. But for the project you should exit the execution
completely.

3) Execution for while loops: first, evaluate the loop condition. If it evaluates to False, the loop
body is guaranteed to be dead code and you should exit with a dead code error, as described
above. If it evaluates to True or AnyBool, execute the loop as was done in Project 4, but do not
re-evaluate the loop condition: just continue executing the loop body until convergence,
similarly to what was done in Project 4.

4) Although this was not discussed in class, your abstract semantics for || and && should use
short-circuit evaluation. Specifically, if the first operand of && evaluates to the abstract value
False, the second operand is not evaluated at all. Similarly, if the first operand of || evaluates
to the abstract value True, the second operand is not evaluated at all. In all other cases, use the
abstracted semantics shown on slide 24.
5) Arithmetic subexpressions of boolean expressions (e.g., \( x+1 \) in \( x+1<y \)) are evaluated as in Project 4, including the errors for division by zero and uninitialized variables from Project 4.

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
```bash
cd p5
make clean
cd ..
tar -cvzf p5.tar.gz p5
```
Then submit p5.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.