USCC
Programming Assignment 0 – Installation (Linux)

By: Sanjay Madhav
University of Southern California

Prerequisites

USCC will work natively on Linux, though you must build via the command line. These instructions assume you are using Ubuntu 14.04. But any similar Linux distribution will work. In addition to Linux, you need to install clang and g++, which you can do with:

```
$ sudo apt-get install clang
$ sudo apt-get install g++
```

You will also need approximately 5GB of free space.

I also strongly recommend you change your default linker to `gold`, because it will link significantly faster than if you use the default `ld` (and it requires much less memory, to boot). On Ubuntu 14.04, `ld` is just a symbolic link to `ld.bfd`, and you want to change that to symbolic link to `ld.gold` instead. You can do this with the following commands:

```
$ cd /usr/bin
$ sudo rm ld
$ sudo ln -s ld.gold ld
```

Note that if you do this, you will not be able to link kernel modules anymore. However, you can always change the symbolic link to point back to `ld.bfd` if you need to.

Installing and Building LLVM

1. Open up a terminal and cd to the directory you want to work in. You will be making multiple directories and links, so it's not recommended to do this in ~.
2. Download the LLVM 3.5 source:
   ```
   $ wget http://llvm.org/releases/3.5.0/llvm-3.5.0.src.tar.xz
   ```
3. Extract the archive:
   ```
   $ tar xJf llvm-3.5.0.src.tar.xz
   ```
4. Rename the directory to `llvm`:
   ```
   $ mv llvm-3.5.0.src llvm
   ```
5. Enter the llvm directory:
   ```
   $ cd llvm
   ```
6. Configure LLVM to build a debug build:
   ```
   $ ./configure CC=clang CXX=clang++ --disable-optimized
   ```
7. Build LLVM (this may take over an hour depending on your machine, but you only have to do this once):
   ```
   $ make
   ```
8. Once you've built LLVM, you need to make a couple of links in the parent directory, so:
   ```
   $ cd ..
   $ ln -s llvm/Debug+Asserts/lib/ lib
   $ ln -s llvm/Debug+Asserts/bin/ bin
   ```
9. LLVM is now setup for use with USCC

Forking and Cloning the Starting Code

1. Open the following URL in your browser: https://bitbucket.org/uscc/uscc-projects
2. Fork the repository *making sure you make your fork private*. Please do not make a public fork as it will allow anyone, including other students, to see your code and potentially copy off of it. For more details, refer to the following Bitbucket documentation page: https://confluence.atlassian.com/display/BITBUCKET/Forking+a+Repository
3. Now go back to the terminal and make sure you are in the directory that contains the `llvm` directory as well as the two symbolic links to `bin` and `lib`
4. Clone your fork into a local directory called `uscc`:
   ```
   $ git clone https://username@bitbucket.org/username/forkname.git uscc
   ```
   (You can get the exact URL in the top right corner of your repo's page on Bitbucket.)

Building via Command Line

1. Enter the `uscc` directory:
   ```
   $ cd uscc
   ```
2. Build uscc:
   ```
   $ make
   ```
3. To verify it worked, enter the tests directory:
   ```
   $ cd tests
   ```
4. In the tests directory, run the following:
   ```
   $ ../bin/uscc -a test002.usc
   ```

   You should then get the following output:
   ```
   test002.usc:16:1: error: Function implementation missing
   {
   ^
   1 Error(s)
   ```
5. Then try running the test suite with the following command:
   ```
   $ python testParse.py
   ```

   You should have 22 out of 23 tests fails.

You are now ready to begin working on PA1.