CSCI 2132
Software Development

Lab 4:

Exploring bash and C Compilation

Instructor: Vlado Keselj

Faculty of Computer Science

Dalhousie University

Lab Overview

- Exploring shell (bash)
- Compiling C programs

Step 1: Login and Lab Setup

- Login to bluenose
- Create `lab4` directory in SVN and submit
- Change your current directory to `lab4`

Step 2: Exporing shell (bash)

- type `echo $SHELL` to see your shell program
- type: `cat /etc/shells`

Step 3: `.bashrc` file

`which rm`

- copy `~/.bashrc` to `bashrc.old`
- copy `bashrc.old` to `bashrc.new`
- add `bashrc.old` and `bashrc.new` to SVN

Editing `bashrc.new` file

- Add the following contents at the end of file and save:

```
umask 077
alias rm="rm -i"
alias mv="mv -i"
alias cp="cp -i"
```

- **Important to enter exactly as shown!**
- Verify the file using: `source bashrc.new`
- Try: `which rm`
- Commit the files to SVN.

- **Optional: If there are no errors,** copy `bashrc.new` to `~/.bashrc`
- **Try to login in another window**
- You can try in another window: `which rm`
- You should get: `/bin/rm`

Step 4: Editing `.profile` File

- similarly to `.bashrc` file, copy `~/.profile` to `profile.old` and to `profile.new`
- Add both `profile.old` and `profile.new` to SVN

- Using emacs (or other editor) edit `profile.new`

```
case `basename $SHELL` in
  sh|jsh)
        . $HOME/.shrc
        ;;
  ksh)
        . $HOME/.kshrc
        ;;
  bash)
        . $HOME/.bashrc
        ;;
esac
```

- Verify using: `source profile.new`

- Commit files `profile.old` and `profile.new` to SVN
- **Optional step: If there are no problems,** copy the file `profile.new` to `~/.profile`
- Using another terminal window check that you can login without problems
- Check in the second window: `which rm`
- Expected output:

```
alias rm='rm -i'
   /bin/rm
```

- Logout from both windows and login again in one window

Step 5: Writing some simple C programs

- Using emacs write `hello.c`
- Add the program to SVN.
- Copy `hello.c` to `hello0.c` and `hello1.c`
- Modify `hello1.c`
- Submit the files '`hello.c`', '`hello0.c`', and '`hello1.c`' to SVN

Step 6: Utility `diff`

6-a) `diff hello.c hello0.c`

6-b) `diff hello.c hello1.c`

6-c) Save the output of 6-b) to `diff.out` and add and commit this file to SVN.

Step 7: Compiling C programs

```
gcc -o hello hello.c
gcc -o hello0 hello0.c
gcc -o hello1 hello1.c
```

- Check permissions of `hello`
- `./hello`
- Compile without `-o hello`, find and run program
- Add the file `hello` to SVN and commit.

Step 8: Using Emacs to compile

8-a) Open `hello.c` using `emacs`

8-b) `M-x compile` Enter

8-c) Modify line to: `gcc -o hello hello.c` Enter

8-d) Check compiler output

8-e) `C-x 1`

8-f) `M-! ./hello`

8-g) `M-x compile` Enter

```
gcc -o hello hello.c && ./hello
```

Step 9: Suspending Emacs

• Use emacs to open `hello.c`

9-a) `C-z`

9-b) Compile and run the program

9-c) Bring emacs back to foreground

Step 10: Examining the Exit Code

• Run the programs '`hello`', '`hello0`', and '`hello1`' and check their exit codes

• Try `ls` with different arguments and check exit code

Step 11: Reading about C functions using `man`

```
man printf
man 3 printf
```

- try with `scanf` as well

Step 12: Experimenting with `printf` **function**

- `%`fm`.`pls — general conversion specification

```
emacs testprintf.c
```

(file content is on the next slide)

```
#include <stdio.h>

int main() {
  int value1 = 123, value2 = 12345;

  printf("[%4d]\n", value1);
  printf("[%-4d]\n", value1);
  printf("[%4d]\n", value2);
  printf("[%-4d]\n", value2);

  return 0;
}
```

- Compile and run the program
- Add file `testprintf.c` to SVN and commit

- Final Notes:

Run SVN commit once more to be sure that all most recent files are submitted to the SVN.

By now, you have finished the required work of this lab.