Static Code analysis hands-on

Getting Started

1. Login into the Linux server
2. Create a new directory called splint in your home directory <home>
   `mkdir` *splint*
3. Go inside the directory you have created in (2) /<home>/splint
   `cd` *splint*
4. Copy the following files from the path as mentioned by the trainer:
   a. sample1.c
   b. sample2.c
   c. sample3.c
   d. sample4.c
   e. sample5.c
   f. sample6.c

```
user72@trainux01:~/splint$ ll
total 32
drwxrwxr-x  2 user72 user72 4096 Nov 23 17:06 ./
drwx------ 25 user72 user72 4096 Nov 23 17:06 ../
-rw-rw-r--  1 user72 user72 1108 Nov 23 16:00 sample1.c
-rw-rw-r--  1 user72 user72  270 Nov 23 16:03 sample2.c
-rw-rw-r--  1 user72 user72 1759 Nov 23 16:47 sample3.c
-rw-rw-r--  1 user72 user72 1219 Nov 23 16:51 sample4.c
-rw-rw-r--  1 user72 user72 1134 Nov 23 17:01 sample5.c
-rw-rw-r--  1 user72 user72 1935 Nov 23 17:06 sample6.c
```

Static Code analysis using Splint

5. Read through the code for sample1.c and statically check the file

   `splint` *sample1.c*

   Closely analyze the warnings given by Splint. Some of the warnings given by a static code analyzer may not be valid for your code.
   E.g. suppose in this example you do not want the warnings related to unused parameters and variables. Try giving the splint command with –paramuse and –varuse to inhibit these warnings:

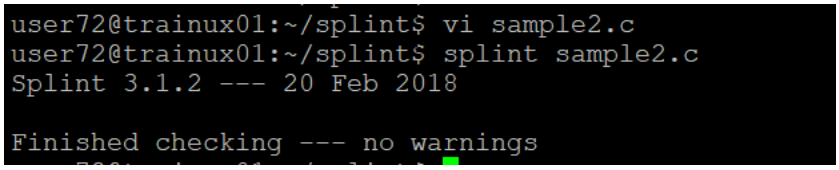
   `splint –paramuse –varuse` *sample1.c*

```
user72@trainux01:~/splint$ vi sample1.c
user72@trainux01:~/splint$ splint sample1.c
Splint 3.1.2 --- 20 Feb 2018

Finished checking --- no warnings
```

6. Read through the code for sample2.c and statically check the file

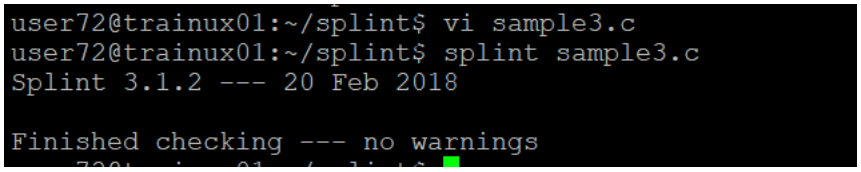
   `splint` *sample2.c*



   Edit this file to fix all the warnings and re-run splint on the updated program

7. Read through the code for sample3.c and statically check the file

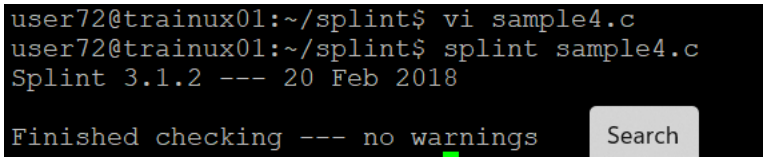
   `splint` *sample3.c*



   Edit this file to fix all the warnings and re-run splint on the updated program

8. Read through the code for sample4.c and statically check the file

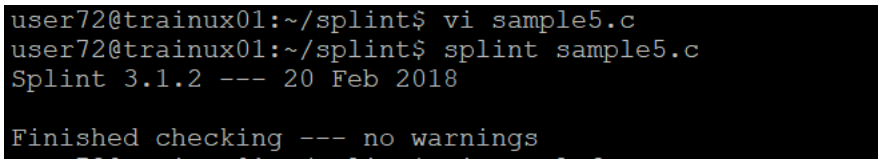
   `splint` *sample4.c*



   Edit this file to fix all the warnings and re-run splint on the updated program

9. Read through the code for sample5.c and statically check the file

   `splint` *sample5.c*



   Edit this file to fix all the warnings and re-run splint on the updated program

10. Read through the code for sample6.c and statically check the file

    `splint` *sample6.c*

```
user72@trainux01:~/splint$ vi sample6.c
user72@trainux01:~/splint$ splint sample6.c
Splint 3.1.2 --- 20 Feb 2018

Finished checking --- no warnings
```

Edit this file to fix all the warnings and re-run splint on the updated program

Including Static Code analysis as part of the makefile

11. Copy the files below to your working directory (which were used in makefile assignment). Create the project directory structure and copy them to appropriate directory. Add a makefile in make directory to include options to run splint tool on files program.c, simplelink.c. Fix the issues reported.

a. program.c
b. simplelink.h
c. simplelink.c

[You may reuse the makefile created earlier and edit to include splint static analysis]