Metso Jouni Antero, 701018-T076, Linux som utvecklingsmiljö, 9th of April, 2017

Reporting Task

All students submit an individual report with these three parts:

Part 1

1. *Function and use*

   Function

   The library, libresistance.so has a function for calculating the total resistance for different connected resistances. The resistors can be connected in series or in parallel.
   The mixed connections (both in series and in parallel in the same schema) is not calculated.
   Use:
   `float calc_resistance(int count, char conn, float *array);`
   where:

- *count*: the number of connected components.
- *conn*: connection in parallel (P) or in series (S) [P | S].
- **array*: pointer to an array of component values of length *count*.
- Value 0 is returned, if any of the values for resistors connected in parallel is zero.
- The library does not crash if zero pointer is sent to the function.
- If arguments are wrong, the function returns -1.
- Otherwise, the function returns the resulting resistance.

   The prototype for the function needs to be included in a C program.

2. *Description of the algorithm.*

   a. The function checks, if array is a null pointer and returns -1 if it is.
   b. The function checks if number of arguments is zero, and returns -1 if it is. This could be defined to zero, if array is not a null pointer, but the arguments must be wrong because length of array disagrees with count. If arguments are wrong, the function must return -1.
   c. Otherwise, there is branch of execution to the parallel part (conn = ‘P’) and the serial part (conn=’S’). The function returns -1 if other function is requested.
   d. Parallel part:
      i. Parallel part checks first, if there is only one non-zero resistance and if is, returns it.
      ii. Then it checks if there is only one resistance and if it is zero. If it is, the function returns -1.
      iii. If any of the parallel resistances is zero, return -1.

iv. Calculate algebraic sum of inverses of resistances and inverse that for equivalent resistance.

e. Serial part:
   i. If there is only one resistor value, return it as serial resistance.
   ii. Otherwise, the equivalent resistance is the sum of *count* resistances in array.
f. Return equivalent resistance, exit function.

3. *How did you compile and test it in your own program. Describe what commands did you use for compiling the library and how do you then use the library in your main program.*

Compilation
Program is compiled with a common Makefile. The commands used are:
gcc: a GNU C compiler
ar: Create archives. Used for creating static libraries.
install: Installs program to directory
I did tests of my own and the test are added as an appendix 1. The test program is available in src/lib1 directory.

There is still a small problem in the code. I the user gives negative count or wrong count compared to the size of array, I didn't find a way to find the wrong values in every situation. This is because C has no way of knowing the real size of the array. The count variable is there because of this. The function uses unsigned variable type. This behaviour (as well as dumping core) is against specifications, because the function was supposed to return -1 then. The flawless function is not guaranteed, if wrong values are used for count.

System and load dependent very large values of array will also cause problems for the library because of memory problems and will dump core. I considered including a mechanism for preventing this by checking the available memory first, but decided that it is probably beyond the scope of this lab. I simply note, that this library implementation is not intended to be used for such a large resistor networks.

I used the library by including header file containing the prototype for the function and then by calling the function.

4. *Describe what switches did you use for the commands above and the function of each switch.*
   Gcc:

- -fPIC for position independent code. This is used for libraries.
- -shared for producing shared library.
- -o creates an executable or library file.
- -c creates an object file (.o).
- -std=c99 is used for lib3 because it uses C99 standard.

- -O sets optimization level.

  Ar:
- r - insert the files on archive (with replacement).
- c - create a new archive
- s - add an index to archive or update it if it exists.

Remove command rm is also used in the Makefile. It uses rf options for removing recursively and ignore nonexistent files and arguments, never prompt.

Part 2

Write a Makefile with these rules:

- *lib, to build only the library.*
- *all, to build both the program and the libraries where libraries are placed in a separate folder, lib, while the man is jus now, for example, / home / bl / electro / lib /. Here, the program will be linked to use the local libraries. NOTE! You can not temporarily change library search path in LD_LIBRARY_PATH!*
- *install. Copy the program and libraries to the appropriate directories (such as / usr / bin / and / usr / lib /) and links so that the program uses the public library.*

Report a description of how you have linked all three libraries and used it in a main program as described above. Also describe what library you worked with.

Describe which commands and switches you use to compile the libraries and then how to use the library in your main program.

All 3 students write the library of their own. Feel free to help each other but do not do the work for each other. - Ask help and help each other. That helps you in writing a common main program and Makefile to compile and use all biblioteken.

Attach an archive of source code and Makefile to compile the project into an executable file.

Svar:
Först så skapar vi alla bibliotek och lägger dem i en undermapp.

lib: lib1 lib2 lib3

```
lib1: src/lib1/libresistance.c src/lib1/libresistance.h
        gcc -c -fPIC src/lib1/libresistance.c -o lib/libresistance.o
        gcc -shared -o lib/libresistance.so lib/libresistance.o

lib2: src/lib2/libpower.c src/lib2/libpower.h
        gcc -c -fPIC src/lib2/libpower.c -o lib/libpower.o -lm
```

```
	gcc -shared -o lib/libpower.so lib/libpower.o -lm

lib3: src/lib3/libcomponent.c src/lib3/libcomponent.h
	gcc -c -fPIC src/lib3/libcomponent.c -o lib/libcomponent.o -lm -std=c99
	gcc -shared -o lib/libcomponent.so lib/libcomponent.o -lm -std=c99
```

Här är alla växlar detsamma som i del 1, med skillnaden att vi har ett extra steg för att skapa delade bibliotek,
-shared skapar ett bibliotek för delad länkning.
För att testa programmet och bygga det så att det använder de lokala biblioteken så använder vi sen följande
del av makelen.

```
all: lib src/electrotest.c
ar rcs lib/libresistance.a lib/libresistance.o
ar rcs lib/libpower.a lib/libpower.o
ar rcs lib/libcomponent.a lib/libcomponent.o
gcc -static src/electrotest.c -Llib -lpower -lresistance -lcomponent -o electrotest_static -lm
```

Här skapar vi på samma sätt som i uppgift 1 lokala arkiv som vi sedan länkar in statiskt i programmet.
Slutligen för att installera biblioteken och programmet så använder vi följande.

```
install: installlib electrotest
install electrotest /usr/local/bin
installlib: lib1 lib2 lib3
install lib/libpower.so /usr/lib
install lib/libresistance.so /usr/lib
install lib/libcomponent.so /usr/lib
electrotest:
gcc -o electrotest src/electrotest.c -lresistance -lpower -lcomponent -lm
```

Här installera vi först de tre biblioteken och länkar sen in dem från /usr/lib. Kör vi nu ldd på det program vi installerade och exekverar det installerade electrotest får vi följande output.

```
axvi@axvi-VirtualBox:~/linum6$ ldd electrotest
linux-gate.so.1 => (0xb77f2000)
libresistance.so => /usr/lib/libresistance.so (0xb77d4000)
libpower.so => /usr/lib/libpower.so (0xb77d1000)
libcomponent.so => /usr/lib/libcomponent.so (0xb77cd000)
libc.so.6 => /lib/i386-linux-gnu/libc.so.6 (0xb7612000)
libm.so.6 => /lib/i386-linux-gnu/libm.so.6 (0xb75c5000)
/lib/ld-linux.so.2 (0x800ab000)
axvi@axvi-VirtualBox:~/linum6$ electrotest
Ange spänningskälla i V: 50
Ange koppling[S | P]: S
Antal komponenter: 3
Komponent 1 i ohm: 300
Komponent 2 i ohm: 500
Komponent 3 i ohm: 598
Ersättningsresistans: 1398.000000
Effekt: 1.788269
```

Ersättningsresistanser i E12-serien kopplade i serie:
1200
180
18

Part 3

Write a brief description of how the collaboration went well as a reflection of what required for this kind of work should work even larger scale with much larger group of developers and major code base.

We had a three person group. One person did not answer to emails and he was replaced. We delivered the tasks and mainly communicated with email. We exchanged Skype information as well.

We created a git repository in github and used gmail in communication. Clear division of work made things easy. Everything went smoothly except for one problem: We had different compiler settings. Library 3 used C99/C11 style variable definitions in loops. My compiler used -std=gnu90 default setting and defining variables inside for-loops triggered error messages. I included compiler switch for newer C-standard in the Makefile and the project compiled for me as well.

In order to work, this kind of code development clearly requires standardised compiler environment and compiler definitions and switched that are unnecessary for most of the developers, but which does not take anything that can change for granted.

Part 2 of the report may as well be identical. Archive uploaded must be of the format tar.gz and named in the following format FormanEfternamn_lab6. tar.gz, such as StureEnoksson_lab6.tar.gz

Appendix 1: Tests

```
xxx@xxx:~/tests/harj6/linum-lab6-g145/src/lib1$ test_libresistance
Parallel resistance with 0 1 ohm resistors with array of 3 1 ohm resistors
 -1.000000
it should return 0  because array is not a null pointer (returns -1 because the parameters are wrong)
Parallel resistance with 0 resistors with null pointer as array
 -1.000000
it should return -1 (ERROR)  because array is a null pointer
Parallel resistance with 2 1 ohm resistors
 0.500000
it should be 0.5
Series resistance with 3 1 ohm resistors
 3.000000
it should be 3
Series resistance with 2 1 ohm and 1 0 ohm resistors
 2.000000
it should be 2
*Parallel resistance with 1 0 ohm resistors
 -1.000000
it should be -1 (ERROR)
Series resistance with 1 5 ohm resistor
 5.000000
it should be 5
Series resistance with 1 5 ohm resistors and count of 2
 5.000000
it should always return -1 (ERROR) but it does not
Series resistance with 1 5 ohm resistors and count of -1
 5.000000
it should always return -1 (ERROR) but it does not
*Parallel resistance with 2 1 ohm and 1 0 ohm resistors
 -1.000000
it should be -1.0 (ERROR)
Parallel resistance with 3 1 ohm resistors
 0.333333
it should be 0.3333
Parallel resistance with 1 5 ohm resistors
 5.000000
it should be 5
Parallel resistance with 1 5 ohm resistors and count of 2
 -1.000000
it should always return -1 (ERROR), but it does not.
**Parallel resistance with 2 ohm resistor 4 ohm resistor and 6 ohm resistor
 1.090909
it should return 1.0909
Parallel resistance with null pointer
 -1.000000
it should return -1 (ERROR)
```