System Programming in C – Homework Exercise 1

Publication date: *Thursday, April 2, 2020*
Due date: *Sunday, April 26, 2020 @ 21:00*

Preliminary preparations: Log into the course server (`sysprog.idc.ac.il`) with your user id and password, create the exercise directory `exercises/ex1/`, and enter it.

If you haven't already changed your password, then do so by using the command `passwd`!!

General guidelines: read carefully before proceeding!!

- Do all your work in the directory `~/exercises/ex1/` that you create for this assignment.
- Problem 1 specifies a list of nine file manipulation tasks for you to implement.
- Each task should be implemented using a single Linux command.
- Use only relative paths; do not use absolute paths. Usage of absolute paths will cause the testing scripts to fail. The only exception to this rule is when you access the shared data directory `/share/ex_data/ex1/` in task #2.

  Tip: if your path starts with / or ~, then it is absolute.
- In Problem 2 you are required to type these commands into a solution file (`my_commands.txt`), which is a file that you submit.
- You should test your solution by following the four validation steps listed on page 3. Your script should pass the final validation step (running `test_ex1`) to guarantee correct submission.
- Follow the submission instructions on page 4. See **Homework submission instructions** for important general details on the submission process for homework assignments in the course.

Problem 1:

1. Create a subdirectory called `dir1` inside the current directory.
2. Copy the contents of the directory `/share/ex_data/ex1` into the newly created directory, `dir1`. After your operation your `dir1` directory should contain two subdirectories (`test_dir1` and `test_dir2`), containing one directory and two files each.
3. Move the two files from subdirectory `test_dir2` into subdirectory `test_dir1`. In this operation, you may assume that files in subdirectory `test_dir2` have names that start with '`ex`' and have an arbitrary extension (i.e. '`.txt`', '`.log`', etc.). After this operation, subdirectory `test_dir1` should contain four files and one subdirectory, and subdirectory `test_dir2` should contain only the subdirectory `dir2`.
4. Move into the subdirectory `test_dir1`.
5. Remove the subdirectory `test_dir2`. Note that this subdirectory is non-empty before you remove it.
6. Change the permissions of the current directory and all the items (files and directories) contained in it and its subdirectories. The new permissions of all items should be: (1) the owner (you) has all permissions (read, write, and execute), (2) users in the group (students) have read and write permissions, but no execute permission, and (3) other users have read and execute permissions, but no write permission. Use an octal permission mode for this.
7. Remove write permissions for yourself from file `assignment1.txt`. Use an alphabetic permission specifier for this.
8. List all items in the current directory whose filename ends with "`2.txt`", and redirect the list into a file named `file.txt`.
9. Use `echo` to add (append) a line containing the text `idc` to `file.txt`.

Problem 2:

Use a text editor (such as `pico`) to type the commands you used in Problem 1 into a text file named `my_commands.txt`. The file should contain exactly 9 lines, each containing a single command. This allows the testing script to test your solution and give you feedback for possible corrections. Make sure that each of the 9 lines in the file contains only the executed command and no additional documentation and labels (which may cause an error while running).

Validation and testing:

1. Verify that file `my_commands.txt` has exactly 9 lines by using command `wc`:
   ```
   ==> wc -l my_commands.txt
   ```
   You will know that you have exactly 9 lines if the result is:
   ```
   9 my_commands.txt
   ```
2. Execute the commands in file `my_commands.txt` as a script using command `source` and examine the contents of the file `file.txt` (using `cat`) and the resulting directory structure (using command `ls`):
   ```
   ==> source my_commands.txt

   ==> cat file.txt
   assignment2.txt
   ex2.txt
   idc

   ==> ls -l ../.. .. .
   .:
   total <SOME NUMBER HERE>
   -r-xrw-r-x 1         <SOME INFO HERE >       assignment1.txt
   -rwxrw-r-x 1         <SOME INFO HERE >       assignment2.txt
   drwxrw-r-x 2         <SOME INFO HERE >       dir1
   -rwxrw-r-x 1         <SOME INFO HERE >       ex1.out
   -rwxrw-r-x 1         <SOME INFO HERE >       ex2.txt
   -rw-r--r-- 1         <SOME INFO HERE >       file.txt

   ..:
   total <SOME NUMBER HERE>
   drwxrw-r-x 3         <SOME INFO HERE >       test_dir1
   -rwxr-xr-x 1         <SOME INFO HERE >       test_ex1

   ../..:
   total <SOME NUMBER HERE>
   drwxr-xr-x 3         <SOME INFO HERE >       dir1
   -rw-r--r-- 1         <SOME INFO HERE >       my_commands.txt
      <YOU MAY HAVE ADDITIONAL FILES HERE THAT YOU CREATED>
   ```
3. Make sure of the following things regarding the script's execution:
   - Ensure that **no error messages** are printed.
   - Run your script from **various locations** (not just `~/exercises/ex1/`) to ensure that it does not contain unwanted absolute paths.
4. Use the script `/share/ex_data/ex1/test_ex1` to test your solution. Execute the following command from directory `~/exercises/ex1/`:
   ```
   ==> /share/ex_data/ex1/test_ex1
   ```
   (the script produces a detailed error report to help you debug your code)

Submission Instructions:

1. After you validated and tested your solution, make sure that your `~/exercises/ex1/` directory contains the `my_commands.txt` script and a `PARTNER` file containing the user id of the non-submitting partner.
2. Only one partner should submit a solution file (`my_commands.txt`). The non-submitting partner should only submit a `PARTNER` file containing the user id of the submitting partner.
3. If you plan to submit your assignment without a partner, you must get explicit permission for this from Sara or Ilan no later than Tuesday, April 21st. Submission without a partner and with no authorization will result in point deduction. If you are submitting alone with permission, specify your own user id in the `PARTNER` file.
4. Check your entire submission (solution file and `PARTNER` file) by running **`check_ex ex1`**. The script should be executed from the account of the submitting partner, and it may be from any directory. The script will provide a detailed report, including the report produced by the test script `/share/ex_data/ex1/test_ex1`. **Clean execution of this script guarantees you 80% of the assignment's grade.**
5. Once you are satisfied with your solution, you may submit it by running **`submit_ex ex1`**. The script should be executed from the account of the submitting partner, and it may be from any directory. The submission script creates a read-only directory `~/exercises/ex1/submit/` containing your solution file and the report from script `check_ex`. You may modify your submission any time before the deadline (**26/4 @ 21:00**) by running **`submit_ex ex1 -o`** from any location.

6. For more information on the submission process, see the **Homework submission instructions** file on the course website.