Gebze Technical University
Department of Computer Engineering - CSE 344 System Programming
Spring 2018-19, HW #2 (Due March 24th @08:30)
NO LATE SUBMISSIONS

This HW is based on the previous one. In this homework, you will create a new process for each directory to get the sizes. Each created processes will be responsible for:

- Finding the size of the directory given by parent process (adding the sizes of subdirectories if -z option is given)(using pipe is forbidden),
- Creating new processes to find the sizes of subdirectories,
- Writing the PID of the process, size(in kilobytes(preferred) or bytes) and path of the directory to a single global file "<#yourstudentid>sizes.txt"(example: 111044002sizes.txt). You must use a file lock as multiple processes shouldn't write to the same file at the same time. Do not remove the file and leave its contents after the execution.

After your child processes write all the size of all directories, main process will read the file, find the total sizes if needed and output to standard output. The order of output is important and it should be post-order. Finally, it will output the number of created child processes and exit.

Do not add the sizes of files or the directories pointed by a symbolic link. Just say that there is a special file. Do not show size of any file explicitly, just directories.

The program `buNeDuFork` with the argument `rootpath` will be called like below. The rootpath can be any path on the system, not just a path in the current directory.

```
./buNeDuFork [-z] rootpath
```

BONUS(20 points): While executing, do not communicate between any process to find total size for -z option, just write your local result to "<#yourstudentid>sizes.txt" file. When all child processes finish, parse that file calculate to total sizes.

When none or meaningless command line arguments are given to the program(s) the output should warn the user and print a usage, informing the user how the program should actually be called.

There shouldn't be any zombie or orphan processes when the execution is finished. Also there shouldn't be any memory leak.

Your homework will be tested by our test directories and files.

Ask your questions in the Moodle forum "HW2 Questions" by opening a new topic.

Example is given in the next page. Your homework should output by the same style given in the example.

Homework format:

Inside zip file→ Your .c and .h files and a makefile named Makefile. You makefile must **just** compile your program when "make" command is given from terminal. There shouldn't be and directory inside zip file.

DO NOT FORGET TO READ ALL OF THE HOMEWORK CAREFULLY AND EXAMINE THE EXAMPLE IN THE NEXT PAGE.

An example:

A (directory)
|- B (directory)
|- kagurachan.exe (15 MB)
|- C (directory)
|- beware(a special file like smybolic link or pipe) (assume 0 KB)
|- squirtle.pu (300 KB)
|- gintoki.png (3 MB)
|- blabla.txt (2 MB)
|- lalala.exe (8 MB)
|- D (directory)
|- beware2(a special file like smybolic link or pipe) (assume 0 KB)
|- okletsgo.sh (1 MB)

Output of “buNeDu -z A” gives total sizes:

PID	SIZE	PATH
1415		Special file beware
1415	300	A/B/C
1414	18732	A/B
1413	0	A/D
1412		Special file beware2
1412	29996	A

4 child processes created. Main process is 1411.

Output of “buNeDu A” don’t add subdirectory sizes:

PID	SIZE	PATH
1415		Special file beware
1415	300	A/B/C
1414	18432	A/B
1413	0	A/D
1412		Special file beware2
1412	11264	A

4 child processes created. Main process is 1411.