GTU

DEPARTMENT OF COMPUTER ENGINEERING

CSE 344 – Spring 2022

HOMEWORK 2 REPORT

SÜLEYMAN GÖLBOL

1801042656

1. REQUIREMENTS

NONFUNCTIONAL REQUIREMENTS

1. Portability → The application should be portable. All computers that have Linux Distro and GCC compiler can run the program. Also, it can be run on Windows when Windows Subsystem for Linux 2 activated.

2. Maintainability → In case of an error occurrence, the system uses perror in order to give feedback on terminal.

3. Performance → The system should initially be able to process as many entries as possible. Each request must be processed with different terminals. The system's performance should be fast enough to show user the feedback.

FUNCTIONAL REQUIREMENTS

In order to compile the program, user have to use "make" command that uses gcc. If make or gcc is not installed user can install it via *"sudo apt-get install build-essential"* command.
Make command runs 2 commands.

```
gcc -Wall main.c sg_process_p.c -lm -o processP
gcc -Wall sg_matrix.c sg_process_r.c -lm -o processR
```

The reason that we are creating 2 executables is because of child – parent relation. In order to get the results we need to run executable like below with input and output paths as command line arguments.

```
./processP -i files/inputFile.dat -o files/outputFile.dat
```

If input file exists and we have permission to save the output file, the executable will run successfully.

2. PROBLEM SOLUTION APPROACH

Firstly, the first big problem for me was the Covariance Matrix calculation.

In order the create covariance matrix I found the size. Because of each child has 10x3 matrix for the coordinates (10 rows for different coordinates and 3 columns for dimensions) the size of the covariance matrix must be 3x3.

For the formula of covariance matrix, first we need to get a matrix of deviation scores: a_1x1, a_1x2, . . . , a_10x3 .

Formula of deviation score matrix:

```
Deviation score matrix: a = A – O * A * ( 1 / n )
```

In the formula above, O shows a matrix full of One's. And the size of it is 10x10 (because of coordinates) to multiply.
After we get the deviation score; if we multiply transpose of the matrix with this matrix, we get a new matrix that is n (size, in our case 10) times bigger than covariance matrix.

And if the divide this value to n (10), we get the covariance matrix.

```
Covariance matrix: a‘ * a / n
```

Example from my input file:

main.c | 3fork_exec.c | inputFile.dat | outputFile.dat

```
1   {sglbl}*[s.golbol2018@gtu.edu.tr]
    UniversityOfPRIMORSKA_UniversitaDelLITORALE_GebzeTechnicalUniversity
```

My input file starts with this line and when my program reads first 10 coordinates (30 bytes);
It gives this output to stdout:

```
Process P is reading files/outputFile.dat
Created R_1 with (123,115,103), (108,98,108), ..., (100,117,46)
```

To calculate covariance matrix, first I needed to get the deviation score matrix.

O * A

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \times \begin{pmatrix} 123 & 115 & 103 \\ 108 & 98 & 108 \\ 125 & 42 & 91 \\ 115 & 46 & 103 \\ 111 & 108 & 98 \\ 111 & 108 & 50 \\ 48 & 49 & 56 \\ 64 & 103 & 116 \\ 117 & 46 & 101 \\ 100 & 117 & 46 \end{pmatrix} = \begin{pmatrix} 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \end{pmatrix}$$

O * A * (1 / n)

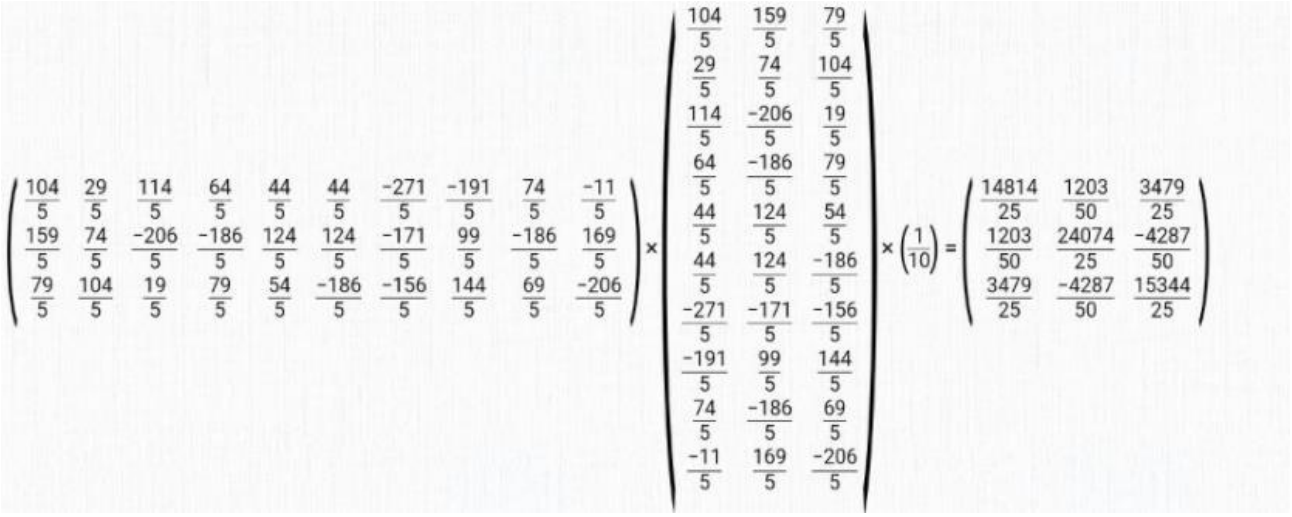
$$\frac{\begin{pmatrix} 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \\ 1022 & 832 & 872 \end{pmatrix}}{10} = \begin{pmatrix} \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \end{pmatrix}$$

a = A – O * A * (1 / n)

$$\begin{pmatrix} 123 & 115 & 103 \\ 108 & 98 & 108 \\ 125 & 42 & 91 \\ 115 & 46 & 103 \\ 111 & 108 & 98 \\ 111 & 108 & 50 \\ 48 & 49 & 56 \\ 64 & 103 & 116 \\ 117 & 46 & 101 \\ 100 & 117 & 46 \end{pmatrix} - \begin{pmatrix} \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \\ \frac{511}{5} & \frac{416}{5} & \frac{436}{5} \end{pmatrix} = \begin{pmatrix} \frac{104}{5} & \frac{159}{5} & \frac{79}{5} \\ \frac{29}{5} & \frac{74}{5} & \frac{104}{5} \\ \frac{114}{5} & \frac{-206}{5} & \frac{19}{5} \\ \frac{64}{5} & \frac{-186}{5} & \frac{79}{5} \\ \frac{44}{5} & \frac{124}{5} & \frac{54}{5} \\ \frac{44}{5} & \frac{124}{5} & \frac{-186}{5} \\ \frac{-271}{5} & \frac{-171}{5} & \frac{-156}{5} \\ \frac{-191}{5} & \frac{99}{5} & \frac{144}{5} \\ \frac{74}{5} & \frac{-186}{5} & \frac{69}{5} \\ \frac{-11}{5} & \frac{169}{5} & \frac{-206}{5} \end{pmatrix}$$

Now if we multiply its transpose matrix with itself and then if we divide to the size we will get covariance matrix.

```
Covariance matrix: a‘ * a / n
```



This value is my covariance matrix of first child.

After this, for the frobenius norm, it wasn’t difficult. I just summed the squares of output values and then get the square root of it.

INTEGER/DOUBLE - STRING CONVERSION

Another big problem that I was into was printing integer or double values using write(STDOUT_FILENO, …) system call.

write() doesn’t have formatter like printf does. So it needs char* as argument and itoa is not C standard function for every environment so I tried to create a int variable and send it’s address to function but it didn’t work while printing so I tried to cast to void* and dereferenced it , also it didn’t work. So I wrote my own itoa function that works for nonnegative ascii integers.

Also I wrote my own double to string function for double value printing. I thought 2 parts of double(before dot, after dot) as different values then I converted them to string with putting “.” between these two.

WAITING

While I was writing my code, parent needed to wait for all the children to finish so I used waitpid in order to success.

```
else if(isFinished == TRUE){
    // Parent process
    if( waitpid(pidCheckIfChild, &status, 0) == -1 ){ // Wait until all children
        if(errno != ECHILD){
            perror("Error on wait() command ");
            exit(EXIT_FAILURE);
        }
    }
    else{
        write(STDOUT_FILENO, "All children are terminated!\n", 30);
        freeingBuffer(buffer, fileSize / (CHILD_SIZE*COORD_DIMENSIONS) + 2 );
        collectOutputFromChildren(argv[4]);  // argv[4] is the output path
        exit(EXIT_SUCCESS);
    }

}
```

Because of that's a system call I also checked errno to make sure that it succeeded. After waiting I freed everything and collected outputs from children.

3) TEST CASES AND RESULTS

3.a) `./processP -i files/inputFile.dat -o files/outputFilePath.dat`

It reads file from files/inputFile.dat path and writes output to the outputFilePath.dat path. Then calculates distance and prints to stdout.

```
gcc -Wall main.c sg_process_p.c -lm -o processP
gcc -Wall sg_matrix.c sg_process_r.c -lm -o processR
sglbl@SglblPC:/mnt/c/Apparatus/GTU/Semester2/CSE344/hw2$ ./processP -i files/inputFile.dat -o files/outputFilePath.dat
Process P is reading files/outputFilePath.dat
Created R_1 with (123,115,103), (108,98,108), ..., (100,117,46)
Created R_2 with (116,114,93), (32,85,110), ..., (110,105,118)
Created R_3 with (101,114,115), (105,116,97), ..., (110,105,99)
Created R_4 with (97,108,85), (110,105,118), ..., (101,121,109)
Created R_5 with (97,110,32), (71,111,108), ..., (98,101,114)
Created R_6 with (32,105,115), (32,49,56), ..., (52,52,44)
Created R_7 with (32,109,121), (32,119,101), ..., (115,103,108)
Created R_8 with (98,108,46), (99,111,109), ..., (117,100,101)
Created R_9 with (110,116,46), (32,73,32), ..., (111,118,101)
Created R_10 with (32,67,32), (98,101,99), ..., (114,111,103)
All children are terminated!
Output file reading finished.
The closest 2 matrices are 9 and 7, and their distance is 6.255
sglbl@SglblPC:/mnt/c/Apparatus/GTU/Semester2/CSE344/hw2$
```

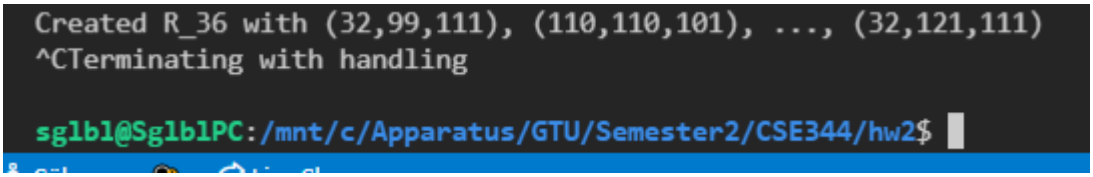
3.b) `./processP Invalid Command Line Argument to Check System Calls`

In the case of invalid command line argument program exits with perror and prints instructions.

```
sglbl@SglblPC:/mnt/c/Apparatus/GTU/Semester2/CSE344/hw2$ ./processP -invalid
Error while opening file. : Bad address
INSTRUCTION: ./processP -i inputFilePath -o outputFilePath
Goodbye!
sglbl@SglblPC:/mnt/c/Apparatus/GTU/Semester2/CSE344/hw2$
```

3.c) `Interrupting with Signals`

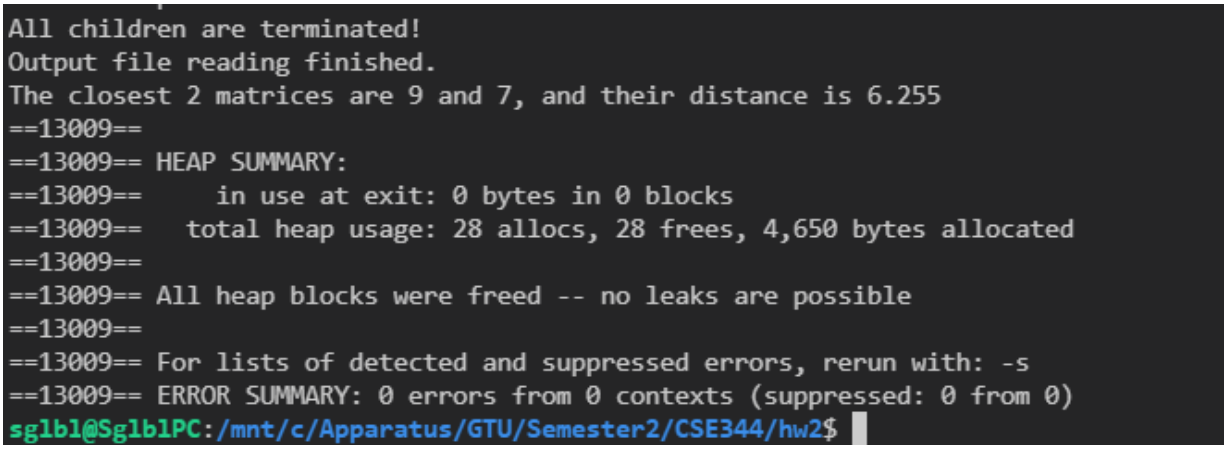
For handling SIGINT signal I used "`static volatile sig_atomic_t`" flag and even though other static/globals are not safe, as official documentation says sig_atomic_t is safe (Reference: https://man7.org/linux/man-pages/man3/signal.3p.html)



When it handles, first it calls all children and children are deallocating their memory and then closing file they've opened and exits. After these, parent deallocates, removes file, do all stuff and then terminates and prints "Terminating with handling" on the screen.

VALGRIND MEMORY RESULTS

The output from valgrind about heap and leaks is like below:



CHECKING FOR ZOMBIE PROCESSES