CSE344 – System Programming - Homework #2 Report

Refik Orkun Arslan
151044063

h2.c (call execve file)

```
88      struct sigaction sa;
        memset(&sa, 0, sizeof(sa));

        sa.sa_flags = 0;
        sa.sa_handler = &handler;                    Implemention Signal Handler

        if(sigaction(SIGINT, &sa, NULL) == -1){
                fprintf(stderr, "err: sigaction\n");
                exit(EXIT_FAILURE);
97      }

99       inputPath = argv[2];                    Assign input/output path
         outputPath=argv[4];
         fd = open(inputPath, O_RDONLY);
         if(fd == -1)
         {
                fprintf (stderr, "\nerrno = %d: %s (open inputPath)\n\n", errno, strerror
                (errno));
                exit(1);
         }
        fprintf( stdout, "Process P reading %s\n",argv[2] );

 109    int processSize=size_file(argv[2])/30;        Filesize/30 byte= how many
        process


136  for(c = 0; c < SIZE; c += 3)      convert unsign byte and add like coordinate
                {
                        sprintf(temp, "(%d, %d, %d), ", (int)((unsigned char) wbuf[c]),
(int)((unsigned char) wbuf[c+1]),(int)((unsigned char) wbuf[c+1])        );
                        strcat(buf1,temp);
                }


                char *bff[]={buf1,NULL};  enviroment assign
            pid_t cur_pid = fork();
```

```
        if(cur_pid == 0) {
            pid[i] = cur_pid;                    Fork and add child process execve
            execve("h2_1",&argv[4],&bff[0]);
            exit(0);
150         }
```

```
157     for(int k = 0; k < i; k++)          wait child process from execve
            wait(&pid[k]);
```

165-210 *Child transactions were written to the file and the file was read.*

```
224     for (int i = 0; i < processSize; i++)
                {           The data from the file was thrown into an array and the closest 2
matrices were found and their lengths were calculated.

                    for (int j = i+1; j < processSize; j++)
                        {

        mtemp=frobeniusNorm(covarianceMatrix[i],covarianceMatrix[j]);
                            if(mtemp<min)
                            {
                                pr1=i;
                                pr2=j;
                                min=mtemp;
                            }
                        }

                }
```

h2_1.c (child processes file)

90-225 Values taken from enviroment variables are discarded in array for covariance matrix calculation

226-275 locking file and write output file

SIGINT Handler

```
struct sigaction sa;
        memset(&sa, 0, sizeof(sa));
```

implementation sigaction

```
        sa.sa_flags = 0;
        sa.sa_handler = &handler;
if(sigaction(SIGINT, &sa, NULL) == -1){
                fprintf(stderr, "err: sigaction\n");
                exit(EXIT_FAILURE);
        }
```

head.h headr file keep inside

```
sig_atomic_t signal_flag = 0;
void handler (int signal_number){
        signal_flag = 1;

}
```

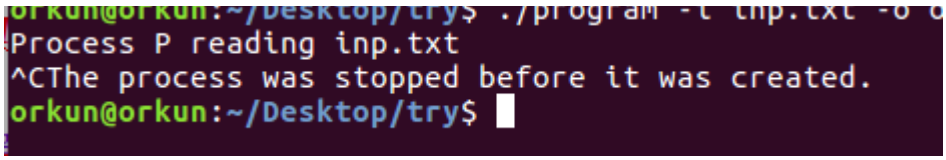
if terminate command comes in parent :

Detect the stop command via (!signal_flag) and check this stop in many places.
EX:

121 while(1 && !signal_flag)
173 if(!signal_flag)
183 while ((len = readline (bfer, 1024, argv[4], &offset)) != -1 && !signal_flag)
218 if(!signal_flag)

it will not run into conditions and loops (when terminated). Child processes will not be created and will go straight to the deallocation.

with the help of spin lock and sleep function



Jumps to the lowest line

if terminate command comes in childs process :

As in the parent process, the signal_flag was used and checked in many places

90 while(i< strlen(envp[0]) && !signal_flag)
226 if(!signal_flag)

if(sigaction(SIGINT,&sa,NULL) == 0) *Use sigavtion signal for to warn parent proces*

fprintf(stdout, "Terminate child procces: %d\n",getpid());
s ignal_flag = 1;

kill(ppid, SIGINT); catch signal and kill parent

***There is no need to kill because the children are terminated**
***when terminate,deallotaion**

```
Process P reading inp.txt
^CTerminate child procces: 19088
Terminate child procces: 19089
Terminate child procces: 19087
orkun@orkun:~/Desktop/try$
```

OUTPUT (NOT TERMİNATE)

inputFile

```
inp.txt (~/Desktop/try) - gedit
Open
5478512354789546987526547856321116156123188946556132484948419999999999999999999999999999999
```

outputFile (write by child processes)

```
Open
5.77778 2.22222 2.22222 2.22222 10.4889 10.4889 2.22222 10.4889 10.4889
4 0.888889 0.888889 0.888889 4.04444 4.04444 0.888889 4.04444 4.04444
0 0 0 0 0 0 0 0 0
```

Final print output

```
Process P reading inp.txt
Created R_2 with (49, 49, 49), (54, 49, 49), (54, 49, 49), (51, 49, 49), (56, 57, 57), (54, 53, 53), (54, 49
, 49), (50, 52, 52), (52, 57, 57), (56, 52, 52),
Created R_1 with (53, 52, 52), (56, 53, 53), (50, 51, 51), (52, 55, 55), (57, 53, 53), (54, 57, 57), (55, 53
, 53), (54, 53, 53), (55, 56, 56), (54, 51, 51),
Created R_3 with (57, 57, 57), (57, 57, 57), (57, 57, 57), (57, 57, 57), (57, 57, 57), (57, 57, 57), (57, 57
, 57), (57, 57, 57), (57, 57, 57), (57, 57, 57),
Reached EOF, collecting outputs from out.txt
The closest 2 matrices are R_1 and R_2 ,and their distance 9.20
orkun@orkun:~/Desktop/try$
```

***No zombie processes were found.**

***No memory leak**

****No terminate problem***

****The output file needs to be cleaned in consecutive runs, otherwise it will give an error.***