System Programming
Homework 2

Muhammed Okumuş
151044107

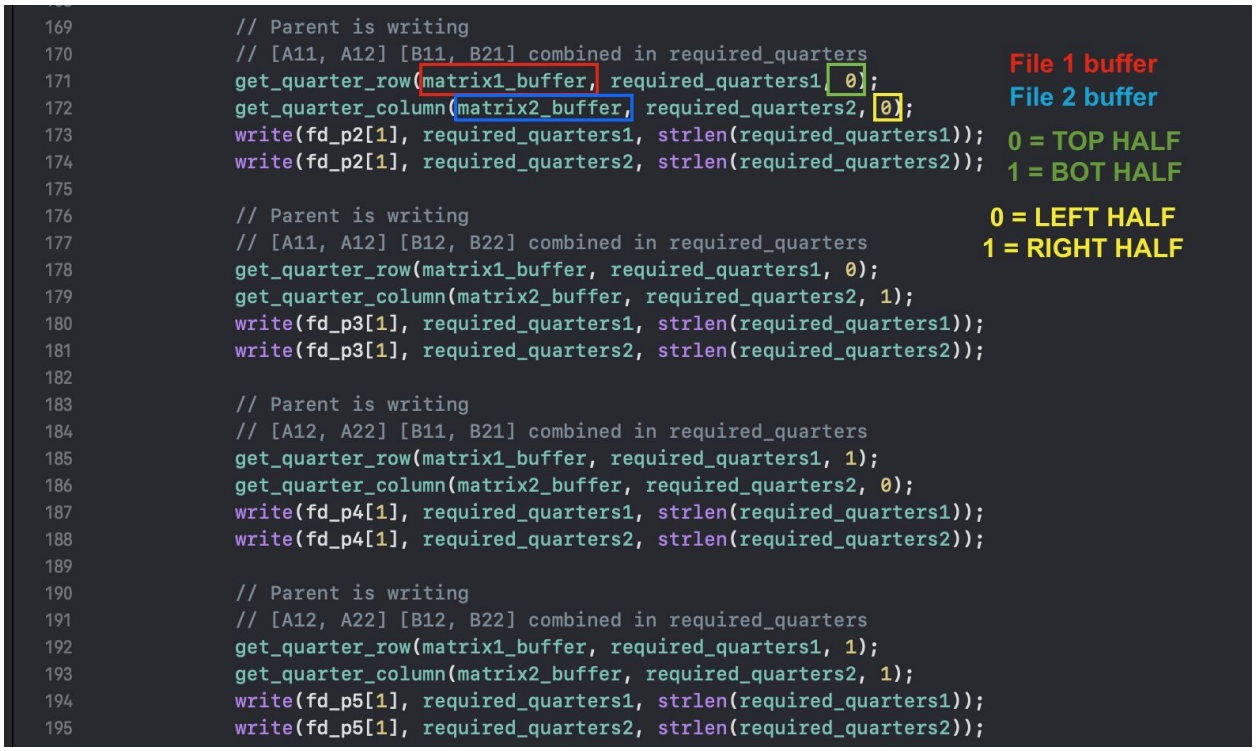
Program Flow

Create 8 pipes, 2 pipes for each child. 1 pipe sending rows and columns of the matrix to the child, 1 pipe for sending the calculated quarter to the parent.

Buffers to be allocated are declared globally so they can be accessed by the exit handler, cleanup(), these buffers are allocated in the parent process after fork. 4 child processes are created by calling fork(). Since the forked process copies all data of the parent process allocating this data before the fork would result in much more memory usage.

In the parent process, validity of input files is checked in the sense of required bytes to fill the matrices. If the file size is enough then it proceeds and allocates buffers to read these files. Also malloc return is checked for each allocation incase of memory unavailability.

After the files are loaded to the buffers, columns and rows required to calculate each quarter are extracted from this buffer and sent to the children processes.



After this the write ends of the pipes are closed and parent waits for all the children process

```
202             // Wait for all the childeren=====================================
203             do{
204                 wpid = wait(NULL);
205             }
206             while (wpid == -1 && errno == EINTR);
207
208             if (wpid == -1){
209                 perror("Wait error\n");
210                 exit(EXIT_FAILURE);
211             }
212             // ====================================Wait for all the childeren
```

After the children are done, parent gathers the input and puts them in a 2D dynamically allocated double array of size $2.N^2 \times N^2$. It has 2 times more rows than it's columns because the SVD function will utilize the bottom half of the matrix.

```
214             // Gather childeren outputs=======================================
215
216             combined_result = malloc(sizeof(double*)*n2*2);
217             for(int i=0;i<2*n2;i++)
218                 combined_result[i] = malloc(sizeof(double)*n2*2);
219
220             singular_values = malloc(n2 * sizeof(double));
221
222             for(int i = 0; i < n2; i++){
223                 for(int j = 0; j < n2; j++){
224                     if(i < n2/2 && j < n2/2) // TOP LEFT
225                         read(cfd_p2[0], &buffer_int, sizeof(int));
226                     else if(i < n2/2 && j >= n2/2) // TOP RIGHT
227                         read(cfd_p3[0], &buffer_int, sizeof(int));
228                     else if(i >= n2/2 && j < n2/2 ) // BOT LEFT
229                         read(cfd_p4[0], &buffer_int, sizeof(int));
230                     else                            //BOT RIGHT
231                         read(cfd_p5[0], &buffer_int, sizeof(int));
232                     combined_result[i][j] = buffer_int;
233                 }
234             }
235
236             // ======================================= Gather childeren outputs
```

After the parent reads from the **pipes they are also closed**. It proceeds to print matrix A(file 1), matrix B(file 2), result Matrix C and its **squared singular values**.

```
238             //Display Matrix A
239             printf("Matrix A:\n");
240             print_matrix(matrix1_buffer, n2, 1);
241
242             //Display Matrix B
243             printf("Matrix B:\n");
244             print_matrix(matrix2_buffer, n2, 1);
245
246             //Display multiplication result
247             printf("Matrix C:\n");
248             display_arr_2d(combined_result, n2);
249
250             //SVD
251             printf("Singular Values Squared:\n");
252             svd(combined_result, singular_values, n2);
253             display_arr(singular_values, n2);
254
```

1 , prints char
0 , prints ascii no

Sample Run

Sample runs are in the project folder named sampeX.txt.