Jacob Schwell
Professor Foreman
CS 311
October 24, 2017

Project 3 Design Document

Summary:
This project reads in two matrices, from either ~/temp/coursein/p3-in.txt on Linux or C:\temp\coursein\p3-in.txt on Windows, and computes their cross product. The matrices only contain single digits and are stored in the document as the first then second separated by a line of asterisks. After reading the values, it determines how many multiplies will be needed and creates a struct and thread for each one. The struct contains the two numbers to multiply and a place to store the product. Each thread is created, passed the struct, and started at the multiple function/subroutine. Once all the threads finish multiplying, they join back up and their products are summed up to form the corresponding elements in the cross product matrix.

Int main():
The Main function opens the matrix input and output files, calls the function to build the matrix, manages the threads and sums the products to form the result thread. Once everything is completed, the main function outputs the cross product matrix to the console and output file, then frees all the data it malloced before and closes the files to ensure no memory leaks are possible. The program was tested with Valgrind to ensure this was the case.

int readMatrixFromFile(FILE* file, int* rows, int* cols):**
This function takes in a file stream--the input file--as well as a pointer to a row and column integer and returns a 2D array of integers. The maximum size of a matrix this function can read in is 128x128. It reads the file on line at a time, then runs through and adds each digit to the next index in the 128x128 matrix, keeping track of the current column. Once the function reads an asterisk or hits the end of file, it knows the matrix is complete. It will then create a new matrix and place the elements from the 128x128 into the new correctly-sized matrix, then return the pointer to it. The row and column values passed, passed by reference, are also updated to the correct values.

void* multiply(void* args):
This function is to be run by an independent thread where void* args is a type-casted struct containing two input numbers and a place to put a result. It simply multiplies the two numbers and places the result in the result element, then returns.

Void freeMatrix(int mat, int rows):**
This function takes a 2D array of integers and the number of rows in order to free each vector and prevent memory leaks.

Examples:

Input:
```
1 1 1 1
1 1 1 1
*******
2 2 2
2 2 2
2 2 2
2 2 2
```

Output:
```
  8    8    8
  8    8    8
```

Input:
```
1 2 3 4 5 6 7 8 9 0
1 2 3 4 5 6 7 8 9 0
******************
9 0
9 0
9 0
9 0
9 0
9 0
9 0
9 0
9 0
9 0
```

Output:
```
 405    0
 405    0
```

Input:
```
1 2 3
3 1 2
2 3 1
*******
9 8 7
7 9 8
8 7 9
```

Output:
```
 47   47   50
  50   47   47
  47   50   47
```