Part 1 - A *(run >make all, then >./A)*

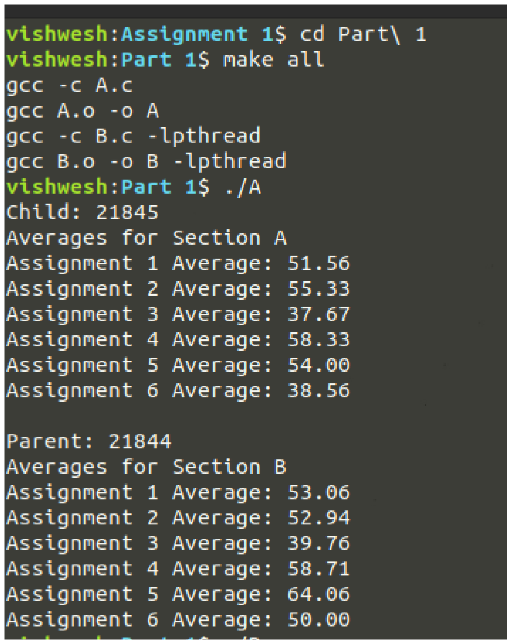
- ➔ Function average() takes the char input of section, calculates and prints the averages.
- ➔ malloc() is used to dynamically allocate memory to store data from CSV file.
- ➔ Error handling to verify if memory is allocated correctly is done.
- ➔ open(), read, close() system calls are used to read the CSV file.
- ➔ Error handling to verify if the file is read correctly is done.
- ➔ Entire file is broken down to lines and each line is parsed using strtok().
- ➔ For correct corresponding assignment and section, the respective sum is incremented.
- ➔ Sums are used to calculate and print averages.
- ➔ Dynamically allocated memory is freed.

- ➔ Function main() creates child process and handles parent and child processes.
- ➔ Child process is created using fork().
- ➔ Error handling to verify if process is created without fail.
- ➔ waitpid() is used to make parent process wait till the child process is complete.
- ➔ Child process calls average() for section A.
- ➔ Parent process calls average() for section B.

Part 1 - B *(run >make all, then >./B)*

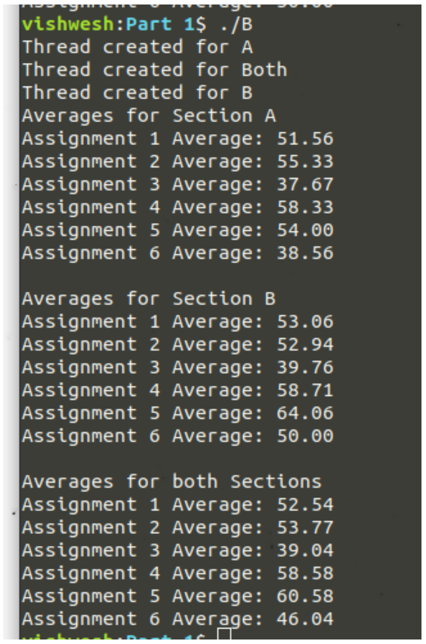
- ➔ Function average() takes the char input of section, calculates and prints the averages.
- ➔ malloc() is used to dynamically allocate memory to store data from CSV file.
- ➔ Error handling to verify if memory is allocated correctly is done.
- ➔ open(), read, close() system calls are used to read the CSV file.
- ➔ Error handling to verify if the file is read correctly is done.
- ➔ Entire file is broken down to lines and each line is parsed using strtok().
- ➔ For correct corresponding assignment and section, the respective sum is incremented.
- ➔ Sums are used to calculate and print averages.
- ➔ Sums and number of students in the particular section is stored globally.
- ➔ Dynamically allocated memory is freed.

- ➔ Function main() creates 3 threads for section A, B and overall average using pthread_create().
- ➔ Thread 1 calls average() for section A.
- ➔ Thread 2 calls average() for section B.
- ➔ Thread 3 is used to calculate overall average using globally stored sum and student count.
- ➔ pthread_join() is used to wait for the threads to terminate.



```
vishwesh:Assignment 1$ cd Part\ 1
vishwesh:Part 1$ make all
gcc -c A.c
gcc A.o -o A
gcc -c B.c -lpthread
gcc B.o -o B -lpthread
vishwesh:Part 1$ ./A
Child: 21845
Averages for Section A
Assignment 1 Average: 51.56
Assignment 2 Average: 55.33
Assignment 3 Average: 37.67
Assignment 4 Average: 58.33
Assignment 5 Average: 54.00
Assignment 6 Average: 38.56

Parent: 21844
Averages for Section B
Assignment 1 Average: 53.06
Assignment 2 Average: 52.94
Assignment 3 Average: 39.76
Assignment 4 Average: 58.71
Assignment 5 Average: 64.06
Assignment 6 Average: 50.00
```



```
vishwesh:Part 1$ ./B
Thread created for A
Thread created for Both
Thread created for B
Averages for Section A
Assignment 1 Average: 51.56
Assignment 2 Average: 55.33
Assignment 3 Average: 37.67
Assignment 4 Average: 58.33
Assignment 5 Average: 54.00
Assignment 6 Average: 38.56

Averages for Section B
Assignment 1 Average: 53.06
Assignment 2 Average: 52.94
Assignment 3 Average: 39.76
Assignment 4 Average: 58.71
Assignment 5 Average: 64.06
Assignment 6 Average: 50.00

Averages for both Sections
Assignment 1 Average: 52.54
Assignment 2 Average: 53.77
Assignment 3 Average: 39.04
Assignment 4 Average: 58.58
Assignment 5 Average: 60.58
Assignment 6 Average: 46.04
```