COMP3230B – Assignment 2 Report

Tesla factory Production Line Control System

Q1: Complete the single threaded version

Ans1: Easily added a few lines of code to job.c to complete 2 functions: makeBattery and makeCar. Created simple threads in main to complete the given 8 jobs.

Q2: Naïve multithreaded program

Ans2: **8-thread version**: Created a function called worker_group that creates a work group of 8 threads and calls the work function in worker.c for the 8 jobs to be completed simultaneously.

Multiple cars: In main.c. num_cars threads are created to call worker_group function using pthread_create. Therefore, there are 2 sets of threads: main threads and worker_group threads.

Q3: Final job

Ans3: **Wasted parts:** I observed that when the num_workers are not a multiple of 8 that is when some group workers not in exact group of 8s, waste parts are created. To handle this problem, I first assigned k jobs (where k<8) to these workers and waited using pthread_join for these k workers to finish their jobs. Then I assigned the remaining jobs to these k workers again until all 8 jobs are completed to make a car.

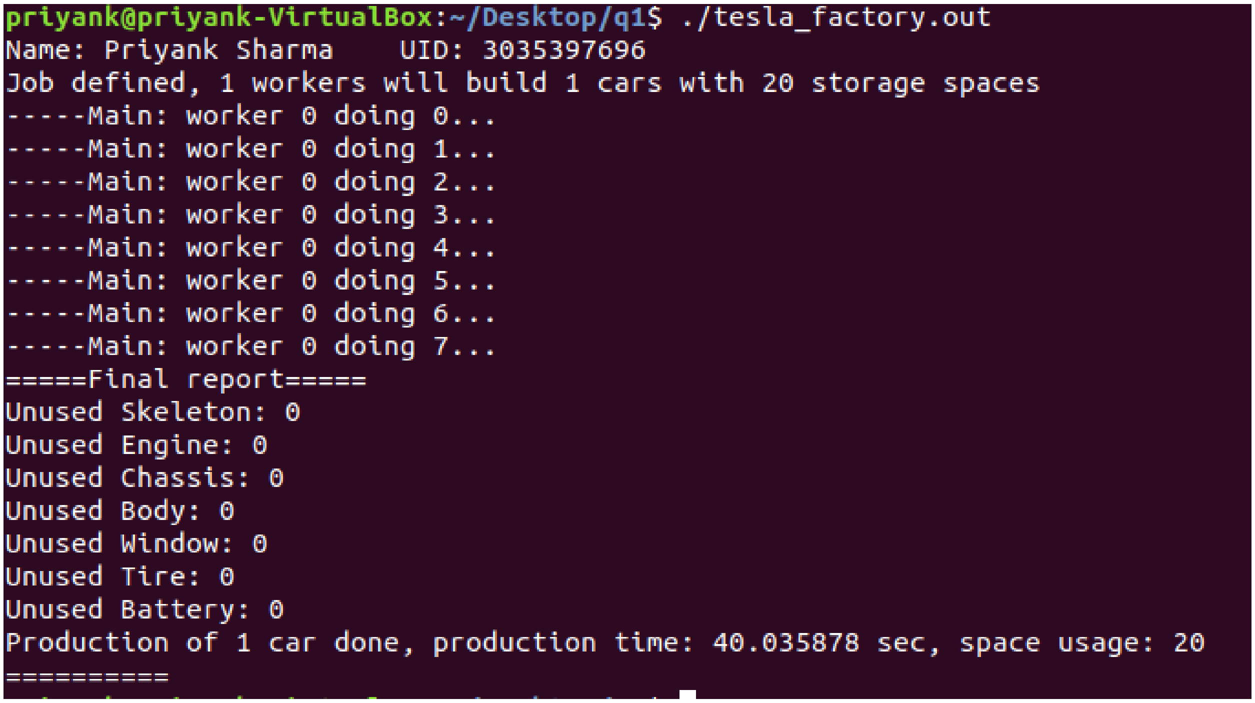
Handle Deadlock: I discovered that at least 16 storage spaces are needed to make one car (7 windows, 4 tyres, 1 battery, 1 body, 1 skeleton, 1 engine and 1 chassis). If the num_spaces is lesser than 16 a separate 'deadlock' code is run which ensures the following:

- Need at least 2 threads/workers to build a car.
- An order of execution of jobs is determined which ensures that the car can be built even with 1 storage space.
- Job 3 that is the body is made first by worker[0]. It waits for the other 3 parts.
- Job 0, 1, 2 makes skeleton, engine and chassis respectively. These are made by worker[1].
- Thread[0] is joined to produce the body.
- Worker[0] now does job 7 that is to make the car. Body is already made so one space is freed up. The car waits for other parts to be made.
- Worker[1] uses this space (and extra spaces if num_spaces>1) to make 7 windows, 4 tires and 1 battery.
- Once all these items are made, thread[1] is joined to produce the car. Since the final product electric car does not take any factory storage space, one space is released for further production.

After making these changes in main.c, I changed the order of getItem in makeCar in job.c. Now first it gets the body, then 7 windows, then 4 tires and finally the battery. This helped to break the deadlock.

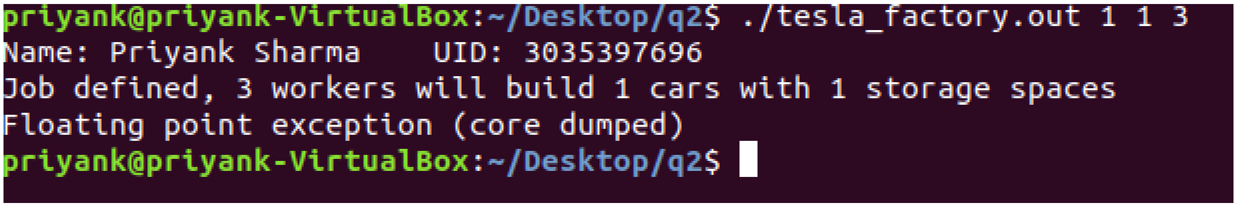
Example code has been provided in the following pages.

Q1:



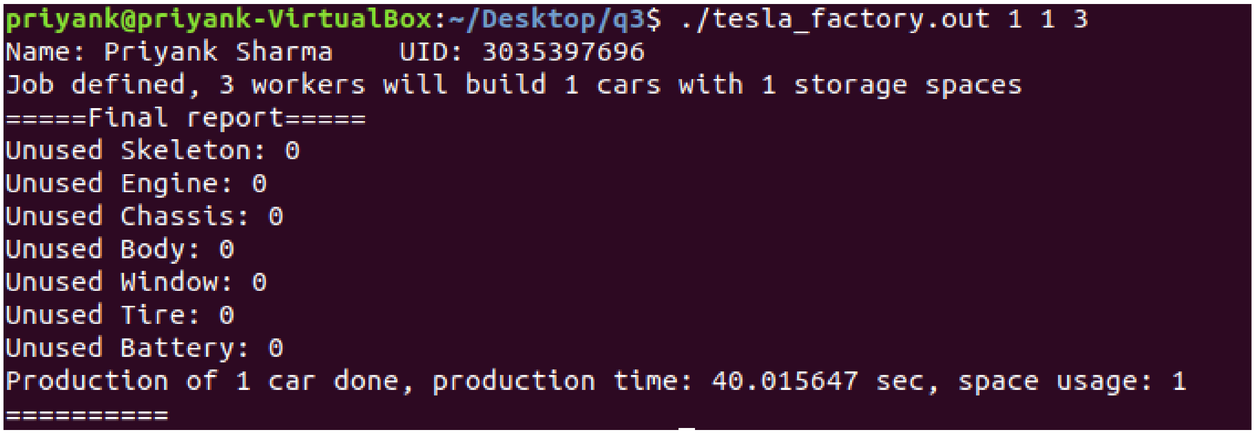
```
priyank@priyank-VirtualBox:~/Desktop/q1$ ./tesla_factory.out
Name: Priyank Sharma    UID: 3035397696
Job defined, 1 workers will build 1 cars with 20 storage spaces
-----Main: worker 0 doing 0...
-----Main: worker 0 doing 1...
-----Main: worker 0 doing 2...
-----Main: worker 0 doing 3...
-----Main: worker 0 doing 4...
-----Main: worker 0 doing 5...
-----Main: worker 0 doing 6...
-----Main: worker 0 doing 7...
=====Final report=====
Unused Skeleton: 0
Unused Engine: 0
Unused Chassis: 0
Unused Body: 0
Unused Window: 0
Unused Tire: 0
Unused Battery: 0
Production of 1 car done, production time: 40.035878 sec, space usage: 20
==========
```

Q3: Without deadlock handler in Q2:



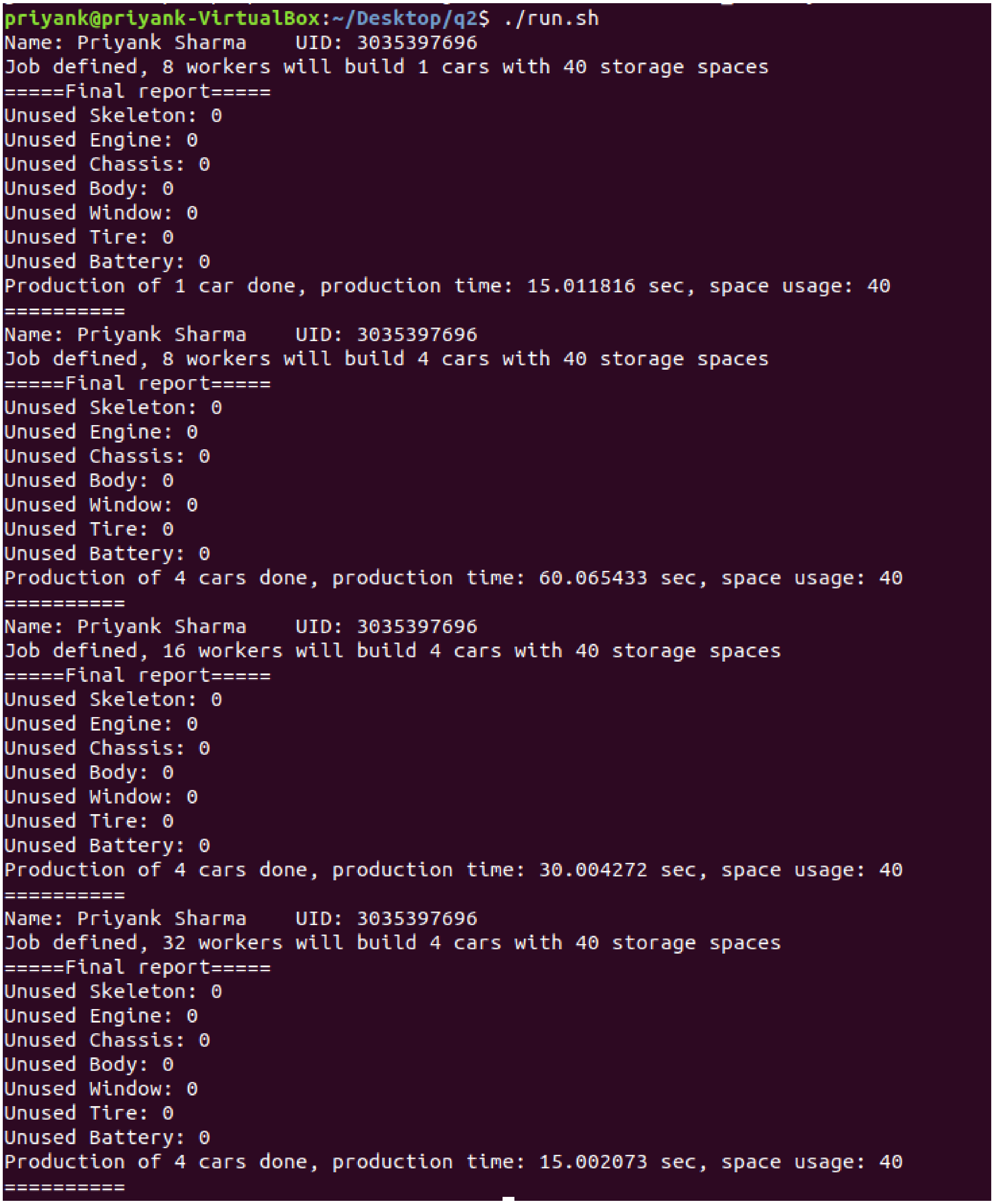
```
priyank@priyank-VirtualBox:~/Desktop/q2$ ./tesla_factory.out 1 1 3
Name: Priyank Sharma    UID: 3035397696
Job defined, 3 workers will build 1 cars with 1 storage spaces
Floating point exception (core dumped)
priyank@priyank-VirtualBox:~/Desktop/q2$
```

With deadlock handler in Q3:



```
priyank@priyank-VirtualBox:~/Desktop/q3$ ./tesla_factory.out 1 1 3
Name: Priyank Sharma    UID: 3035397696
Job defined, 3 workers will build 1 cars with 1 storage spaces
=====Final report=====
Unused Skeleton: 0
Unused Engine: 0
Unused Chassis: 0
Unused Body: 0
Unused Window: 0
Unused Tire: 0
Unused Battery: 0
Production of 1 car done, production time: 40.015647 sec, space usage: 1
==========
```

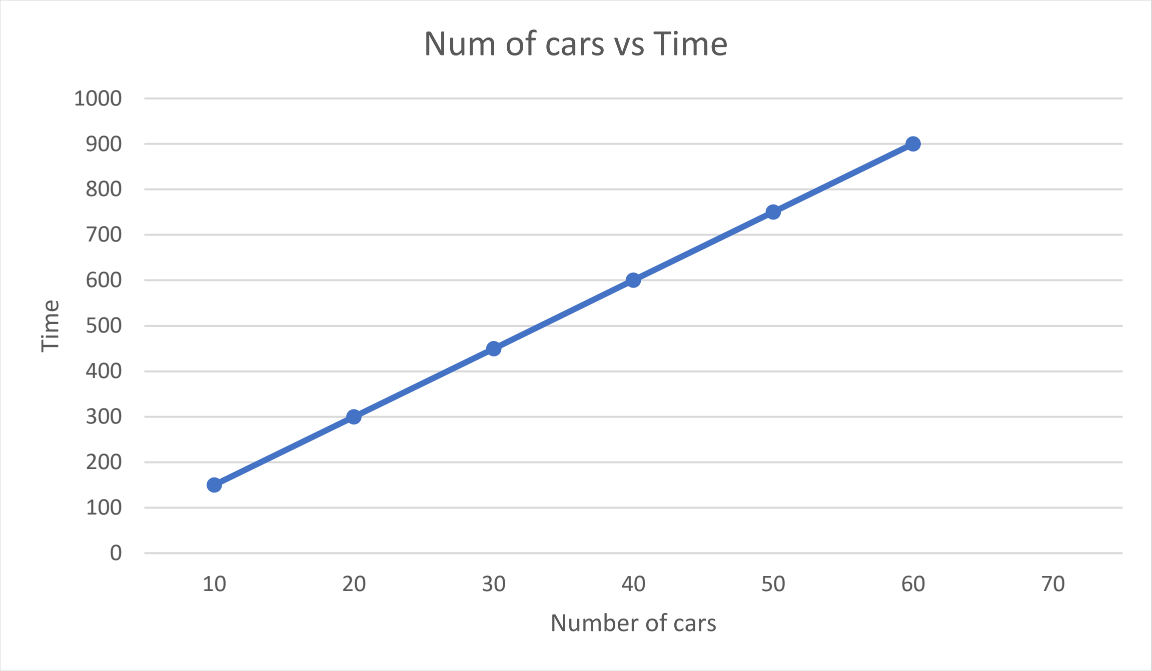
Q2:



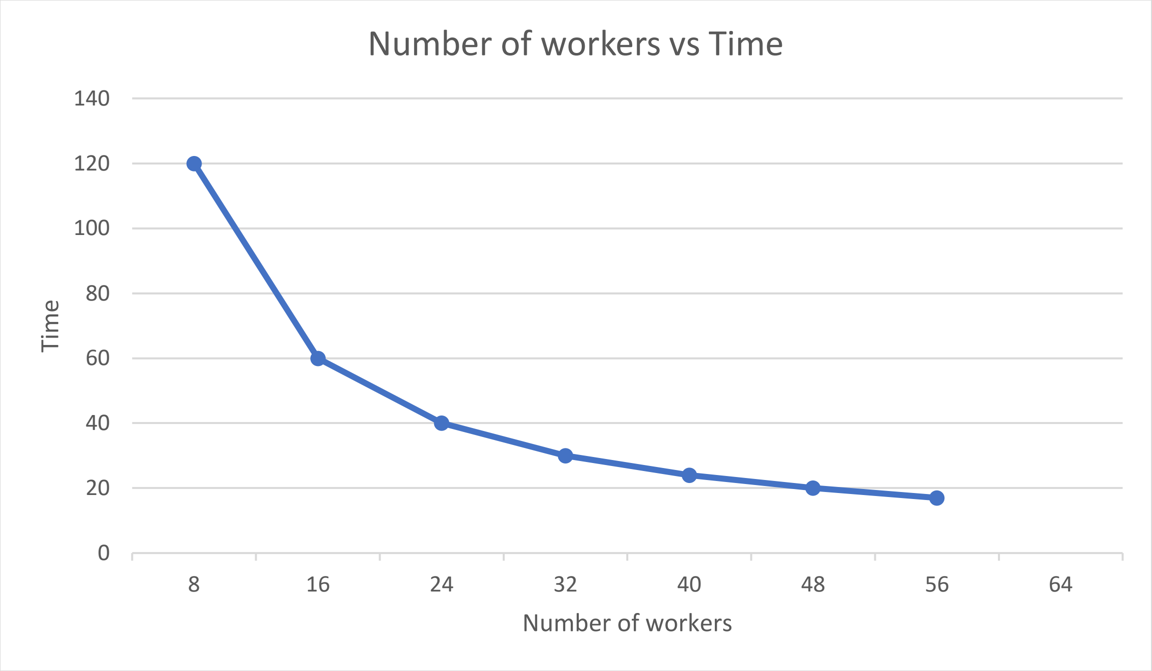
```
priyank@priyank-VirtualBox:~/Desktop/q2$ ./run.sh
Name: Priyank Sharma    UID: 3035397696
Job defined, 8 workers will build 1 cars with 40 storage spaces
=====Final report=====
Unused Skeleton: 0
Unused Engine: 0
Unused Chassis: 0
Unused Body: 0
Unused Window: 0
Unused Tire: 0
Unused Battery: 0
Production of 1 car done, production time: 15.011816 sec, space usage: 40
==========
Name: Priyank Sharma    UID: 3035397696
Job defined, 8 workers will build 4 cars with 40 storage spaces
=====Final report=====
Unused Skeleton: 0
Unused Engine: 0
Unused Chassis: 0
Unused Body: 0
Unused Window: 0
Unused Tire: 0
Unused Battery: 0
Production of 4 cars done, production time: 60.065433 sec, space usage: 40
==========
Name: Priyank Sharma    UID: 3035397696
Job defined, 16 workers will build 4 cars with 40 storage spaces
=====Final report=====
Unused Skeleton: 0
Unused Engine: 0
Unused Chassis: 0
Unused Body: 0
Unused Window: 0
Unused Tire: 0
Unused Battery: 0
Production of 4 cars done, production time: 30.004272 sec, space usage: 40
==========
Name: Priyank Sharma    UID: 3035397696
Job defined, 32 workers will build 4 cars with 40 storage spaces
=====Final report=====
Unused Skeleton: 0
Unused Engine: 0
Unused Chassis: 0
Unused Body: 0
Unused Window: 0
Unused Tire: 0
Unused Battery: 0
Production of 4 cars done, production time: 15.002073 sec, space usage: 40
==========
```

Scalability analysis:

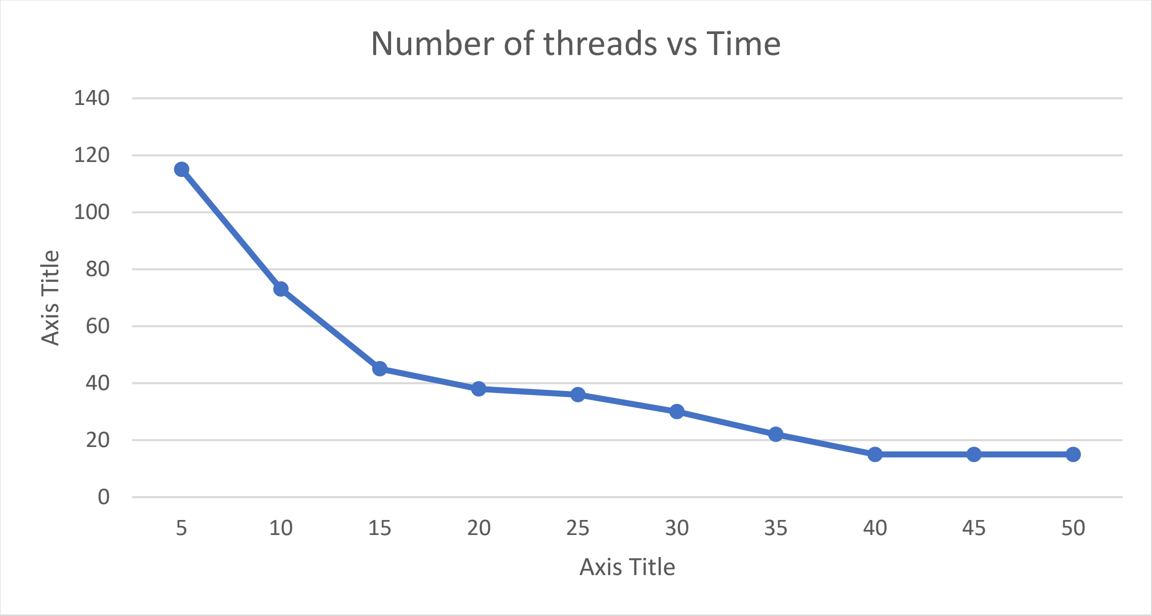
8 Workers and 40 space: There is a proportional correlation here. Given a fixed number of workers and storage space, more cars will take more time to build.



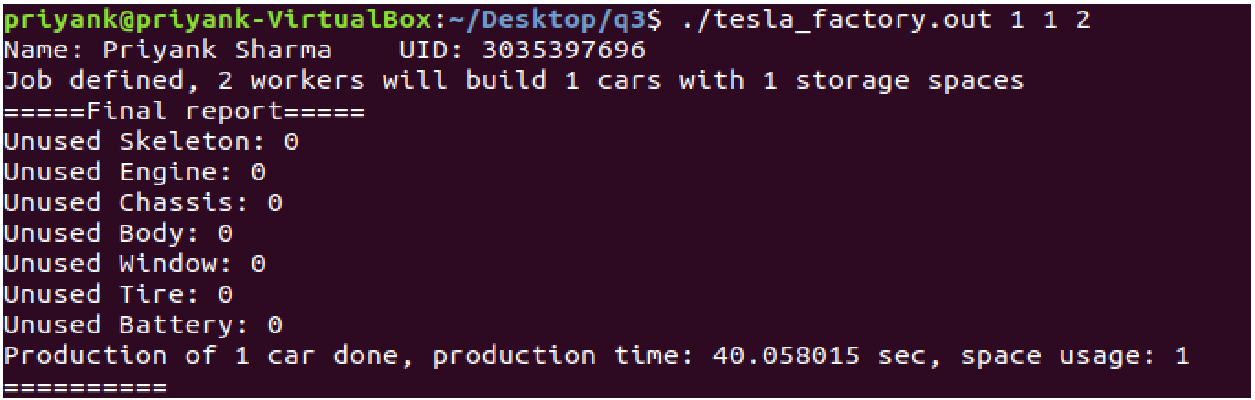
8 cars and 40 space: There is an inverse correlation such that given the number of cars and storage space, more workers mean more threads and hence more tasks will be undertaken. Therefore, more workers result in lesser time for car production.



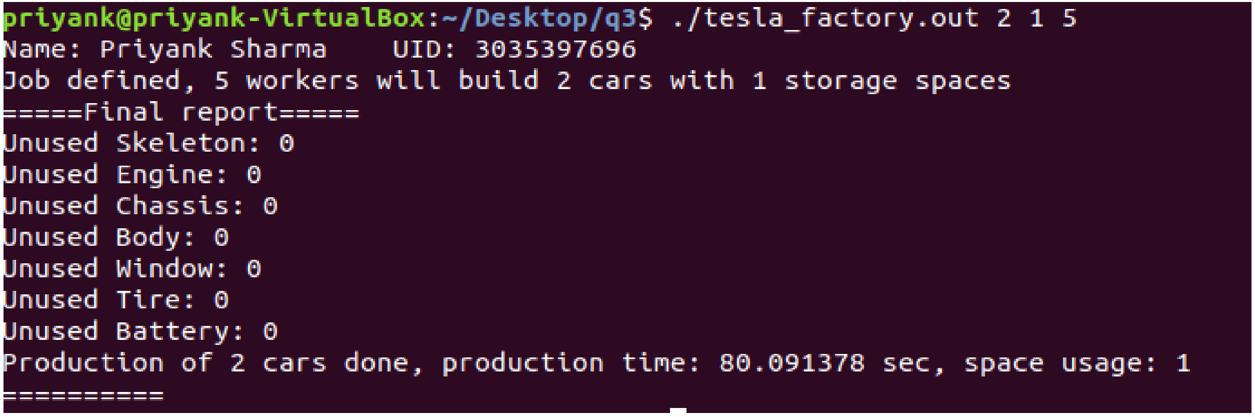
5 cars and 100 space: Given a certain number of cars and enough storage space, fewer threads take more time for car production. As the number of threads increases, time required decreases and then it becomes a constant to 15 seconds since each car does not need more than 8 threads to complete and these 8 threads take maximum of 15 seconds working simultaneously.



Q3: Deadlock handled by code:



```
priyank@priyank-VirtualBox:~/Desktop/q3$ ./tesla_factory.out 1 1 2
Name: Priyank Sharma    UID: 3035397696
Job defined, 2 workers will build 1 cars with 1 storage spaces
=====Final report=====
Unused Skeleton: 0
Unused Engine: 0
Unused Chassis: 0
Unused Body: 0
Unused Window: 0
Unused Tire: 0
Unused Battery: 0
Production of 1 car done, production time: 40.058015 sec, space usage: 1
==========
```



```
priyank@priyank-VirtualBox:~/Desktop/q3$ ./tesla_factory.out 2 1 5
Name: Priyank Sharma    UID: 3035397696
Job defined, 5 workers will build 2 cars with 1 storage spaces
=====Final report=====
Unused Skeleton: 0
Unused Engine: 0
Unused Chassis: 0
Unused Body: 0
Unused Window: 0
Unused Tire: 0
Unused Battery: 0
Production of 2 cars done, production time: 80.091378 sec, space usage: 1
==========
```