**Batman**

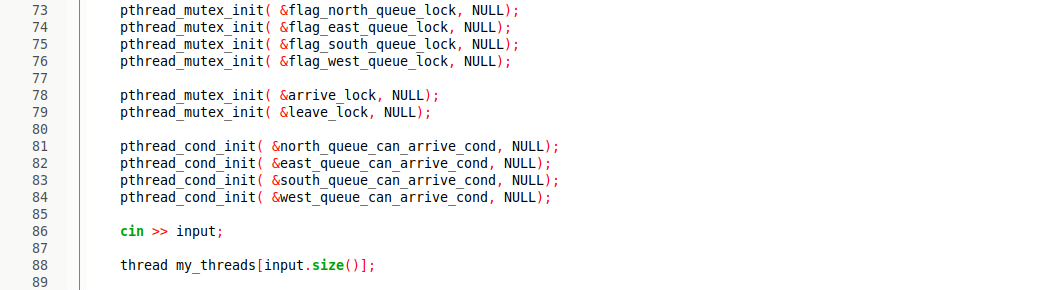
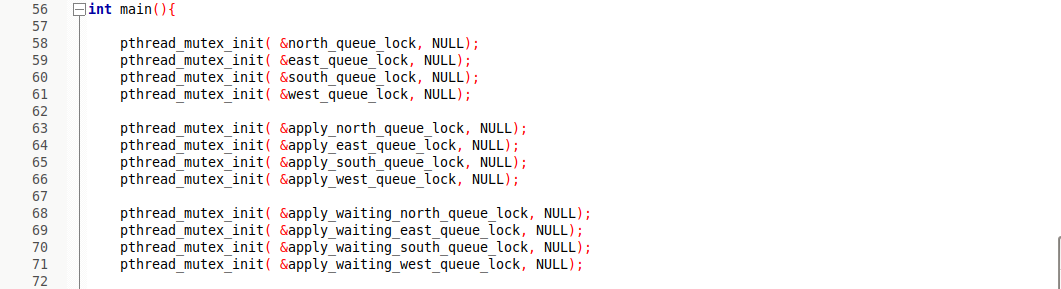
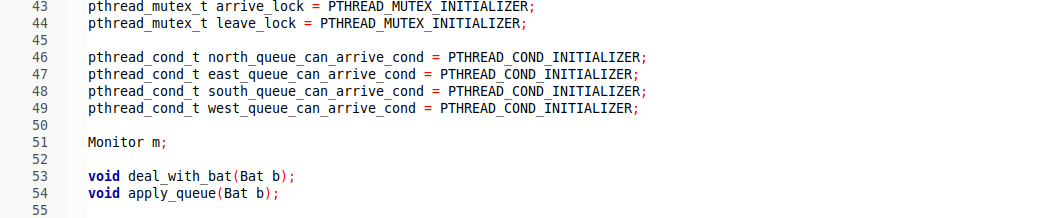
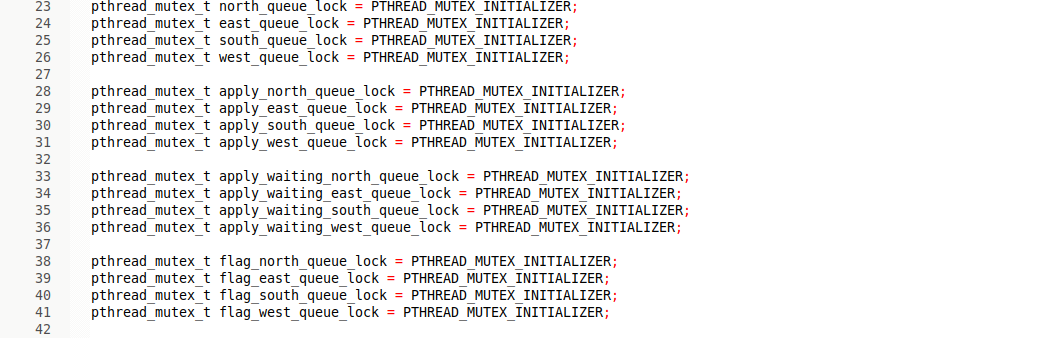
**1-Overall organization and functions:**

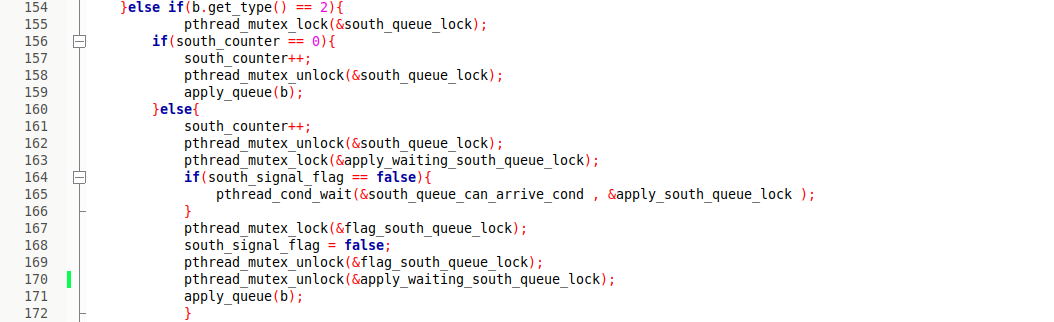
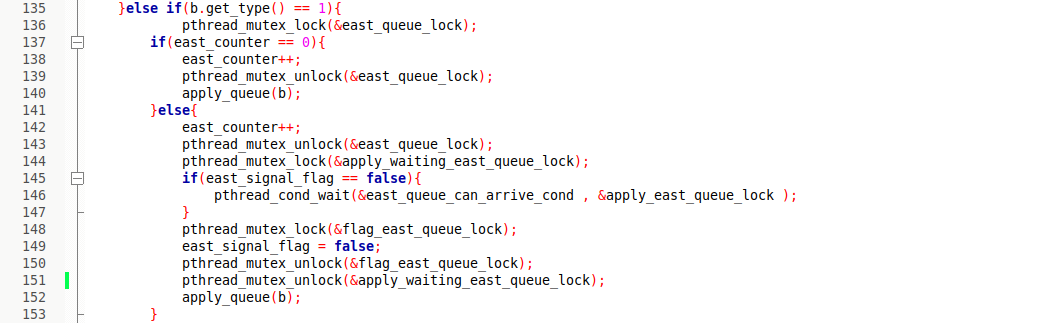
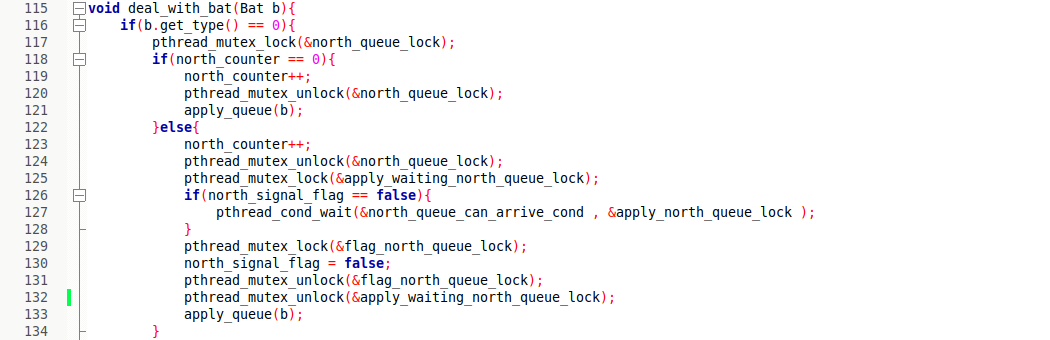
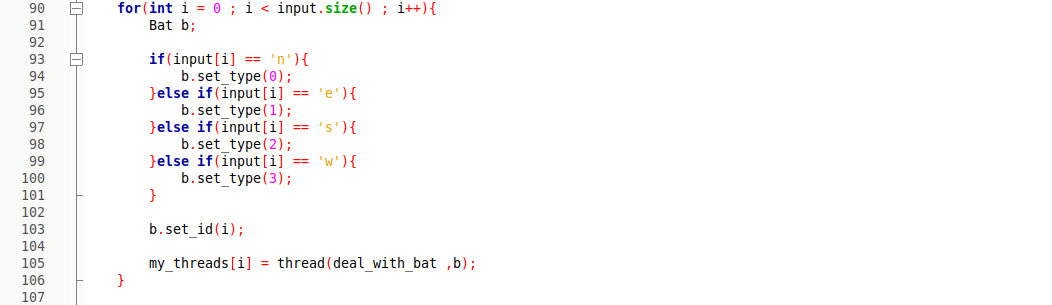
The project is divided into 3 main classes which are : Main , Bat , Monitor

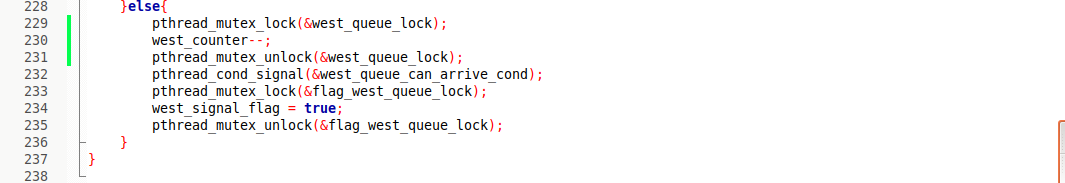
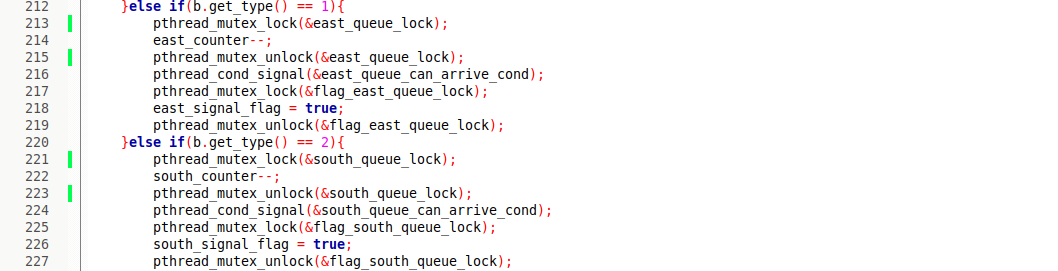
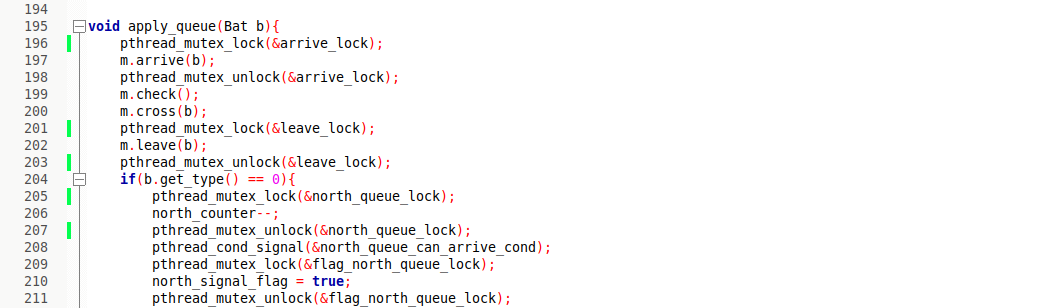
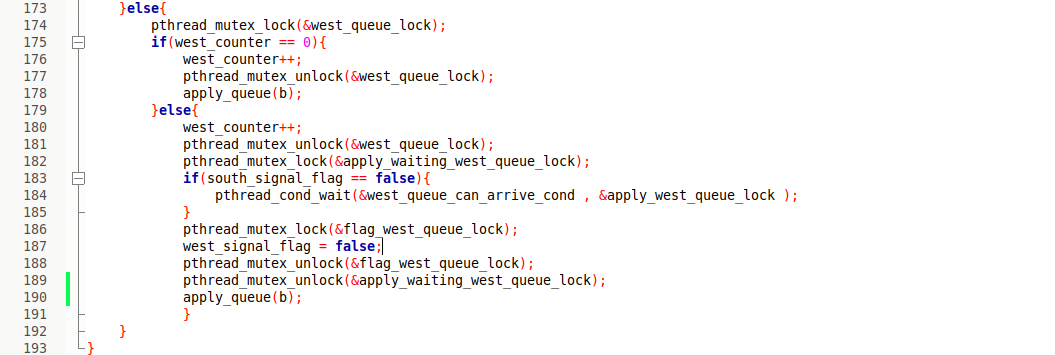
1. Bat class : a class which works like the bat which holds the type of the bat (north , east , …) in integer values ( north : 0 , east : 1 , south : 2 , west : 3 ) and holding an id which indicates it’s index in the input



1. Main class : a class which do most of the work. It starts with initializing a lot of locks and conditions which will be mentioned later , it also take in the input and make a loop for each character and build a bat for each character using the bat class and then in starts a thread which calls a method called deal\_with\_bat which determine the type of the bat and check some conditions which will be mentioned later and then call another method called apply\_queue which start to call arrive , cross , leave by turn and also will be mentioned later. We can divide the class as follow :





* Locks : 6 families were used as the follow

1. north\_queue\_lock : used to make sure that only one thread is checking the value of the north counter and changing it
2. apply\_north\_queue\_lock : used with the waiting condition
3. apply\_waiting\_north\_queue\_lock : used in the waiting case where there is another bat from the same member in the monitor so all other bats will wait and only one will receive the signal to go
4. flag\_north\_queue\_lock : used to make sure only one thread is checking the flag value and changing it.
5. arrive\_lock : used to make sure that only one thread is applying arrive mode now
6. leave\_lock : used to make sure that only one thread is applying leave now

“The last 2 locks are only used to sync the printing to avoid a print of something like : BAT BAT 2 from North arrives at crossing 3 from East arrives at crossing”

* Conditions : 1 family was used as follow

1. north\_queue\_can\_arrive\_cond : used to signal that a thread of a family has left the crossing and another one can come in the crossing

“The bat is only allowed to be in after a member has finished the leave method and this can be modified only by changing the signal place”

* Counters : 1 family is used as follow

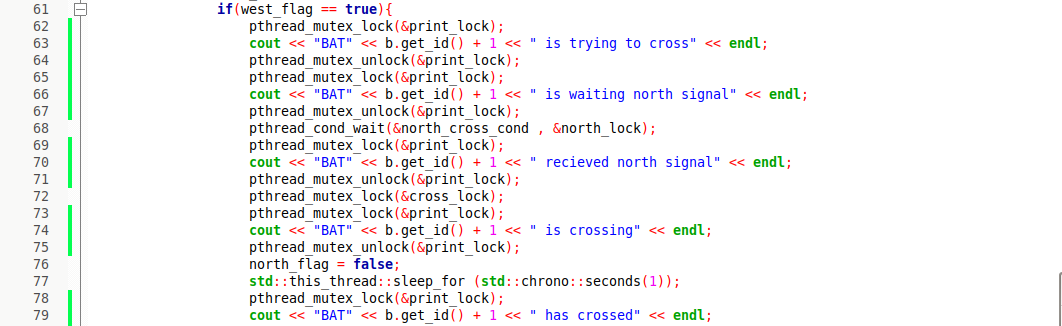
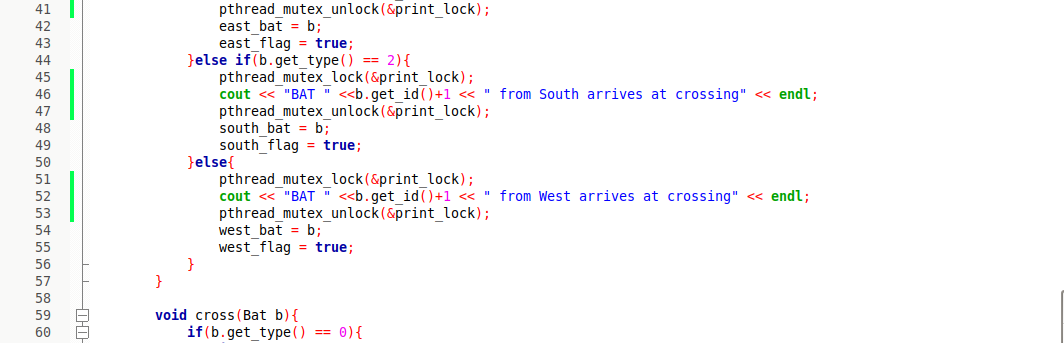
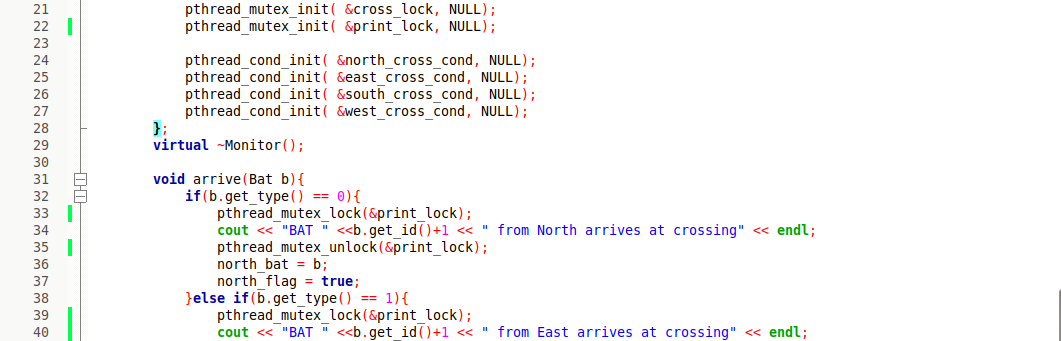
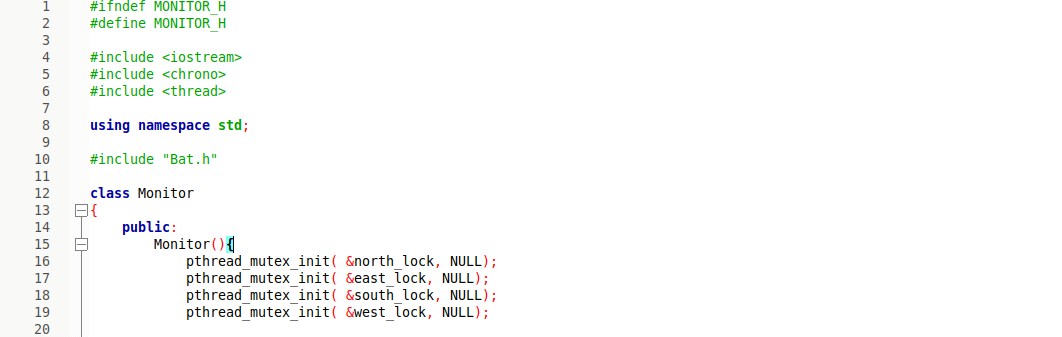
1. north\_counter : used to know the number of waiting threads from the same family

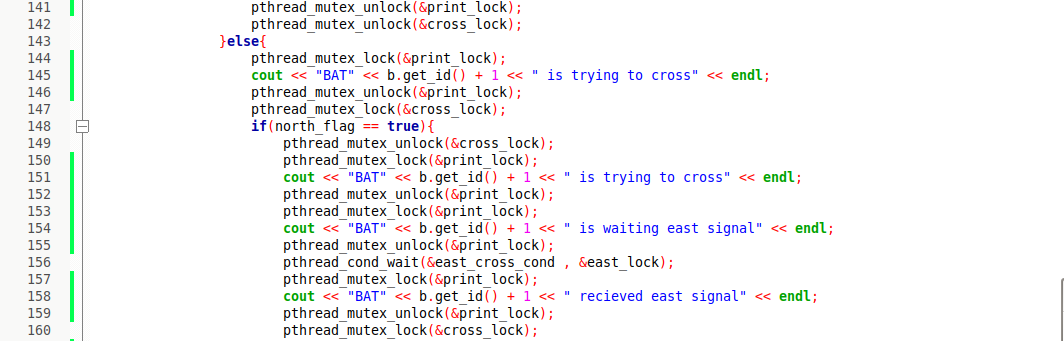
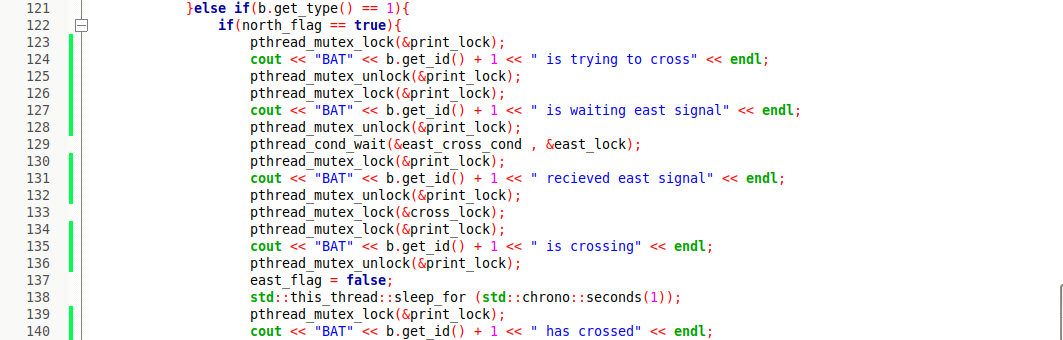
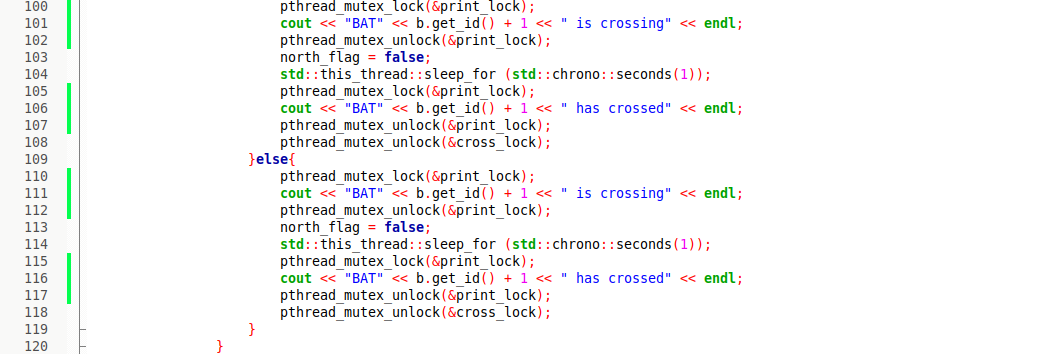
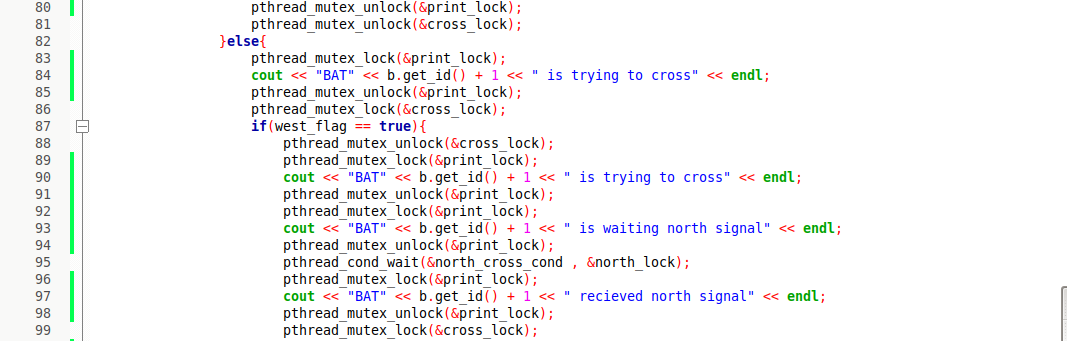
* Flags : 1 family is used as follow

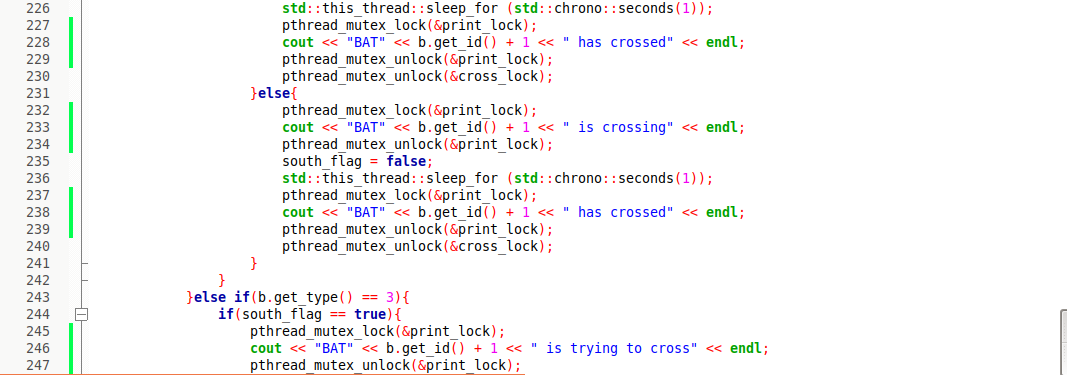
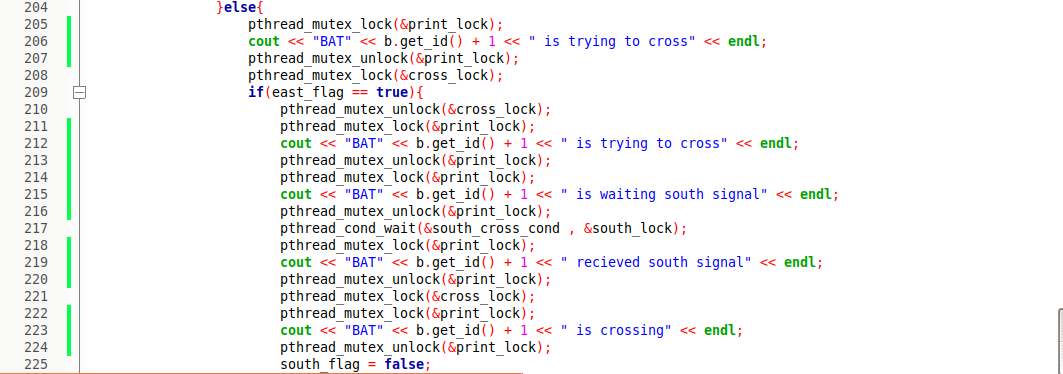
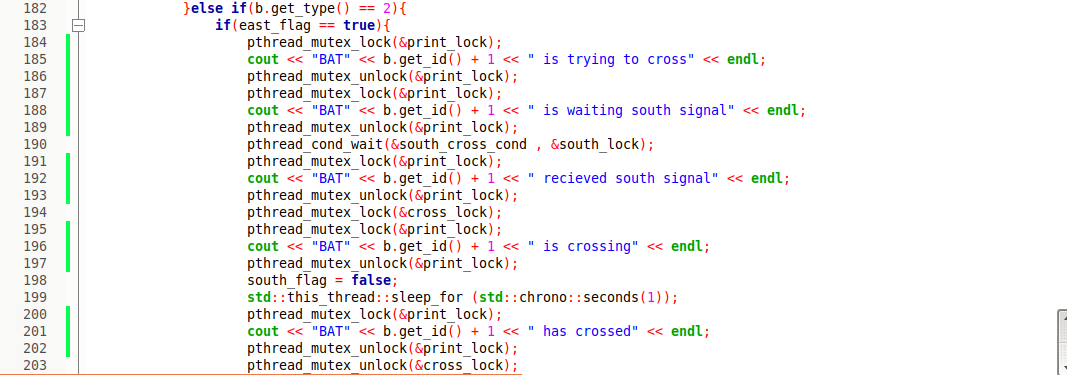
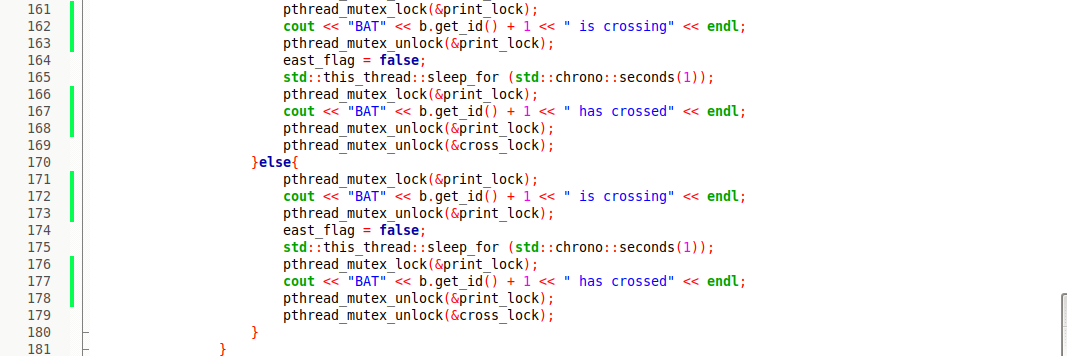
1. north\_signal\_flag : used to solve a problem which occurs. The problems was as follow processor starts with thread A so it goes in to arrive then the processor switches to thread B from the same family so it checks the counter and in founds it should wait but before applying the wait on signal the processor switches again to A and finishes it so it signals a signal but B wasn’t yet waiting on it so it was lost and when the processor switches back to B it was like a deadlock case

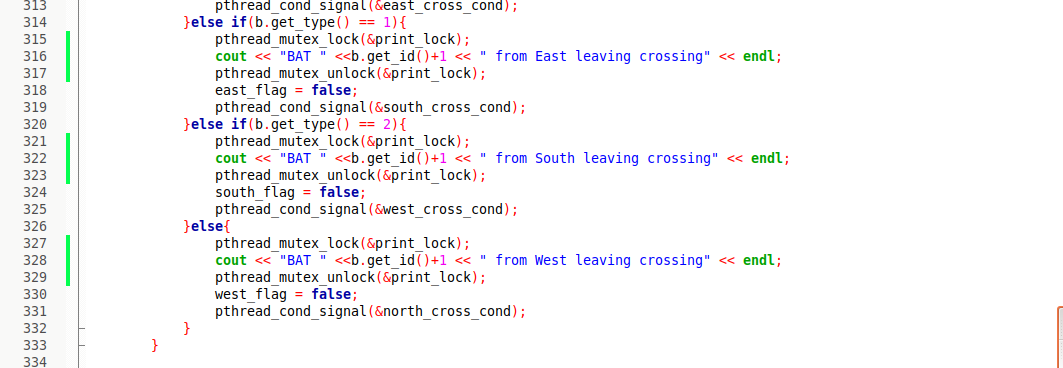
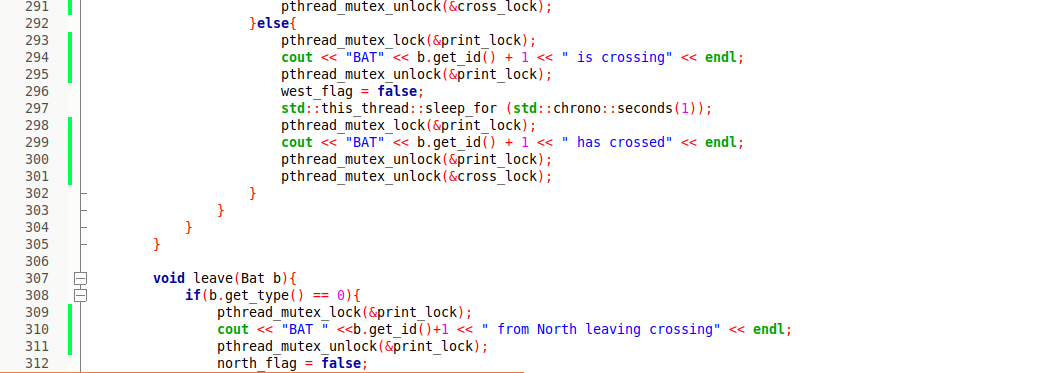
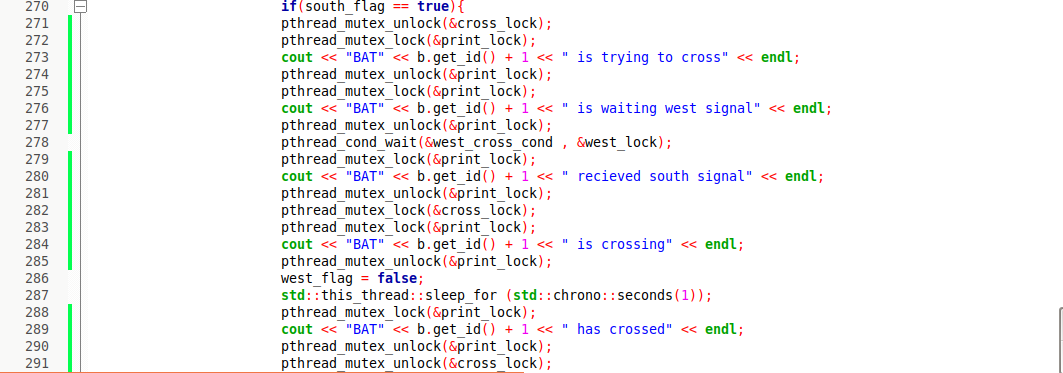
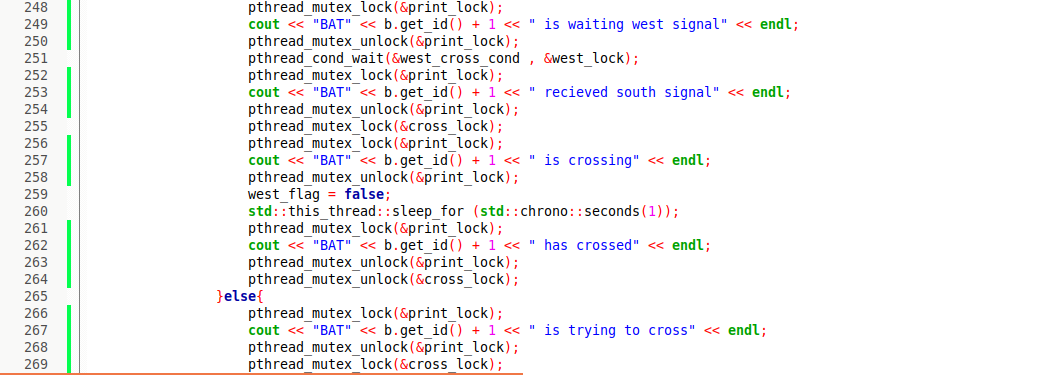
* deal\_with\_bat method : this method is used to make sure that is only one member of the family is in the monitor. It works at first by determining the type of the bat then it locks the counter (for mutual exe) and checks it then unlock it and here it can find that there is no any member from the family in the monitor so it can call apply\_queue method which pushes it to the monitor or it can find that it should wait so it locks the waiting and wait on the signal after checking that there was no lost signals and then it lock the flag and changes it and unlock them all and go to apply\_queue method
* apply\_queue method: this method is working like the navigation for the threads in the monitor which starts with locking the arrive the call arrive then unlock it and then it checks for deadlocks by calling check then cross and then locks the leave and leave and then unlock it after that it locks the counter change it then unlock it finally it signal that any waiting on can move on and set the flag to true to make sure it isn’t lost

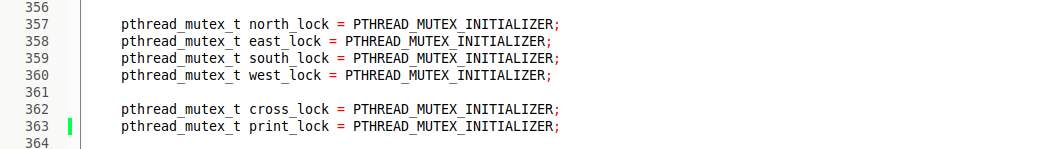
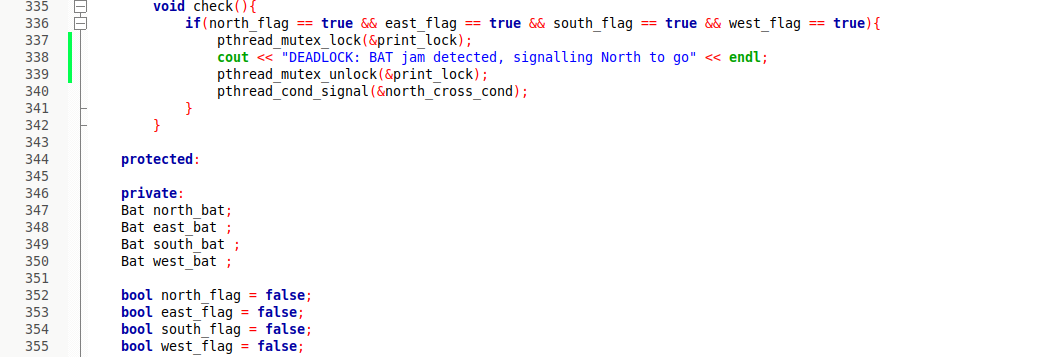
4. Monitor class : this class is very simple as we are sure that we have max of 4 threads from different families at a time so it has 4 objects of bats one for each family and another 4 flags to indicate if there is a member of the family in the monitor or not and 4 signals for the right condition and a print lock and a cross lock we can divide It as follow:





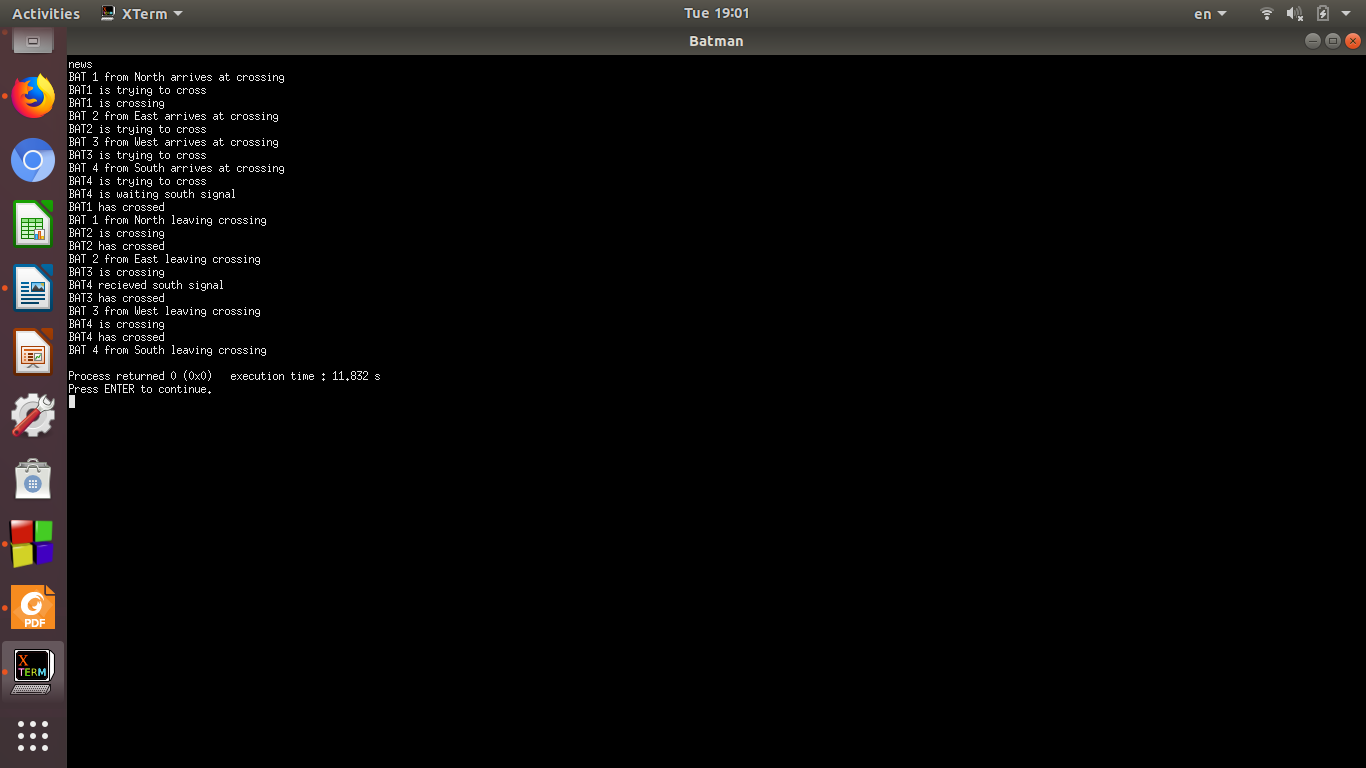




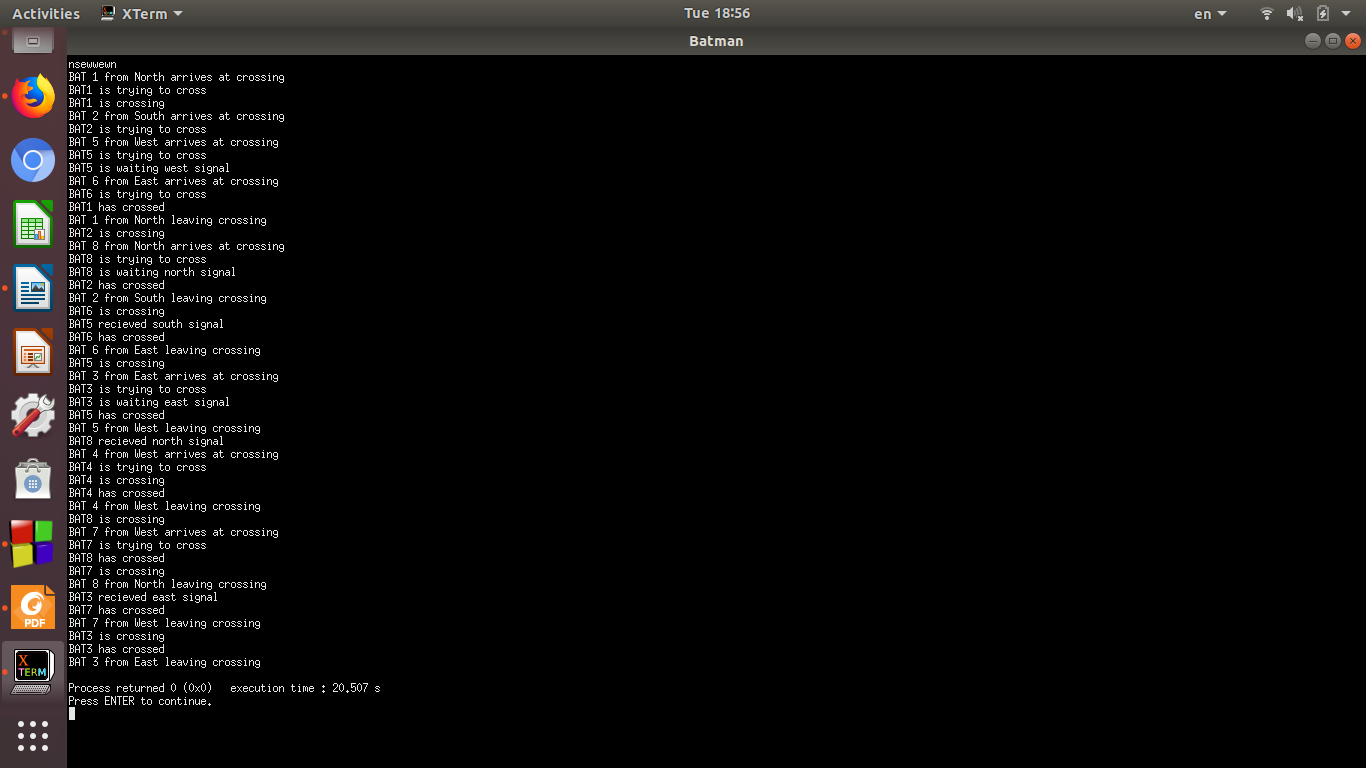


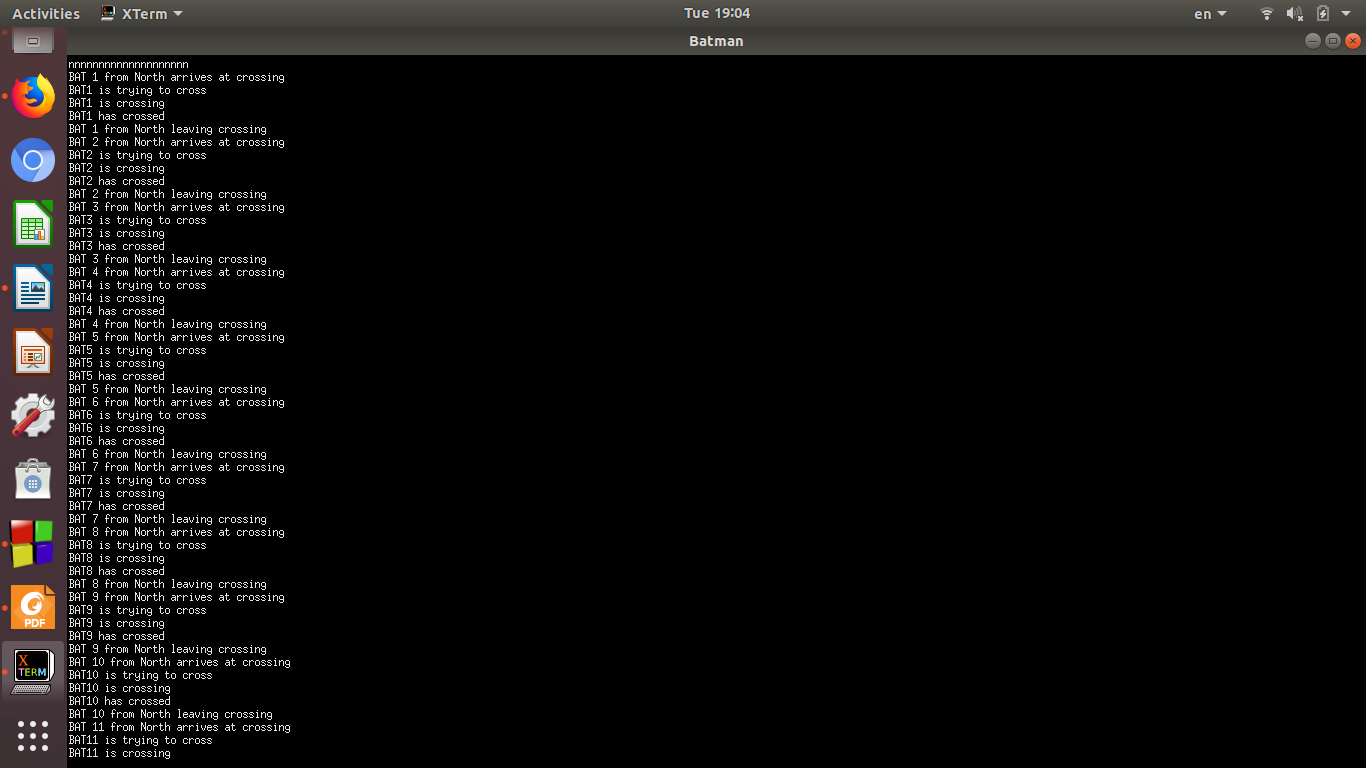
* arrive method : this method print the statement and then set the object to the new one and sets it’s flag
* cross method : this method checks if the flag of the right bat is true and if so it let the right bat to cross then it crosses but if it was false it crosses if the cross is free
* leave method : this method print the statement and then set the flag to false and signals the left to pass
* check method : this method checks the 4 flags after every arrive and if they were all true then a deadlock occur

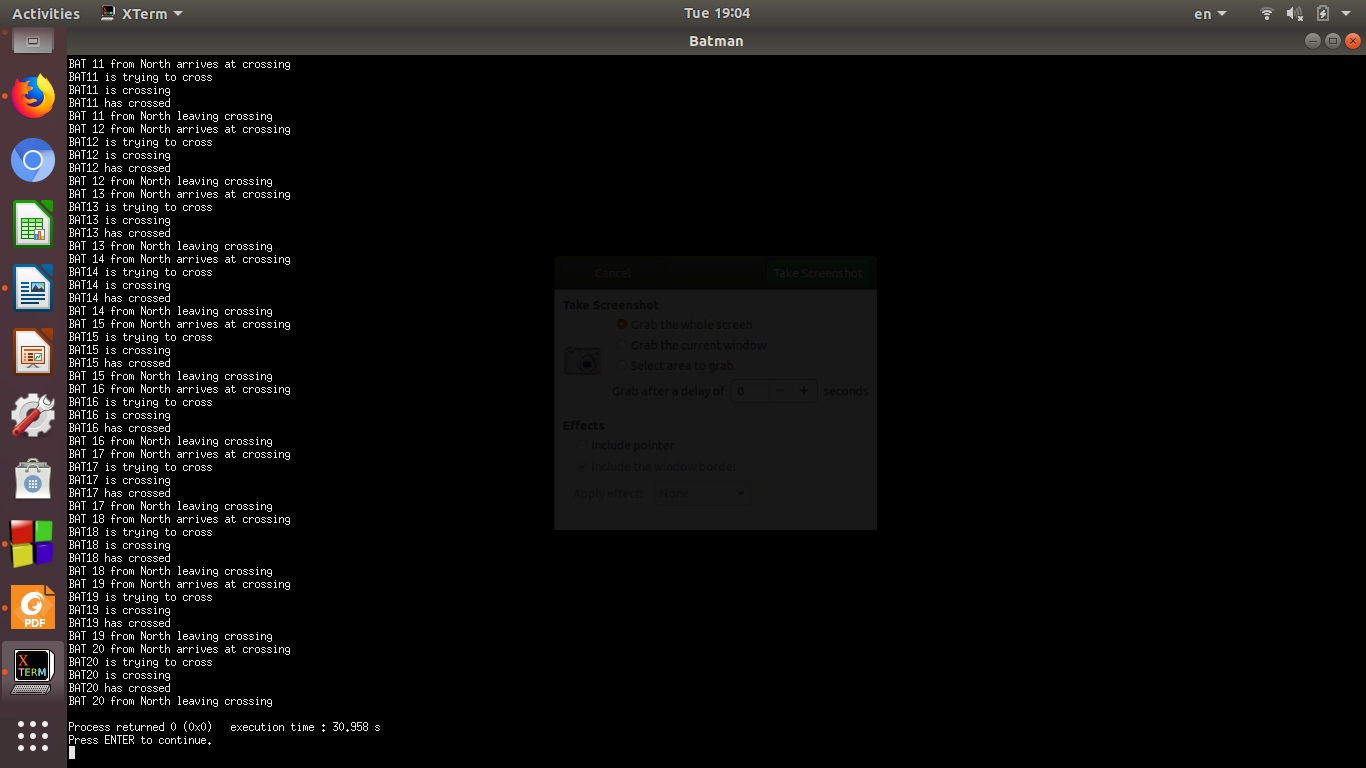
**1-Sample runs and screen shots:**

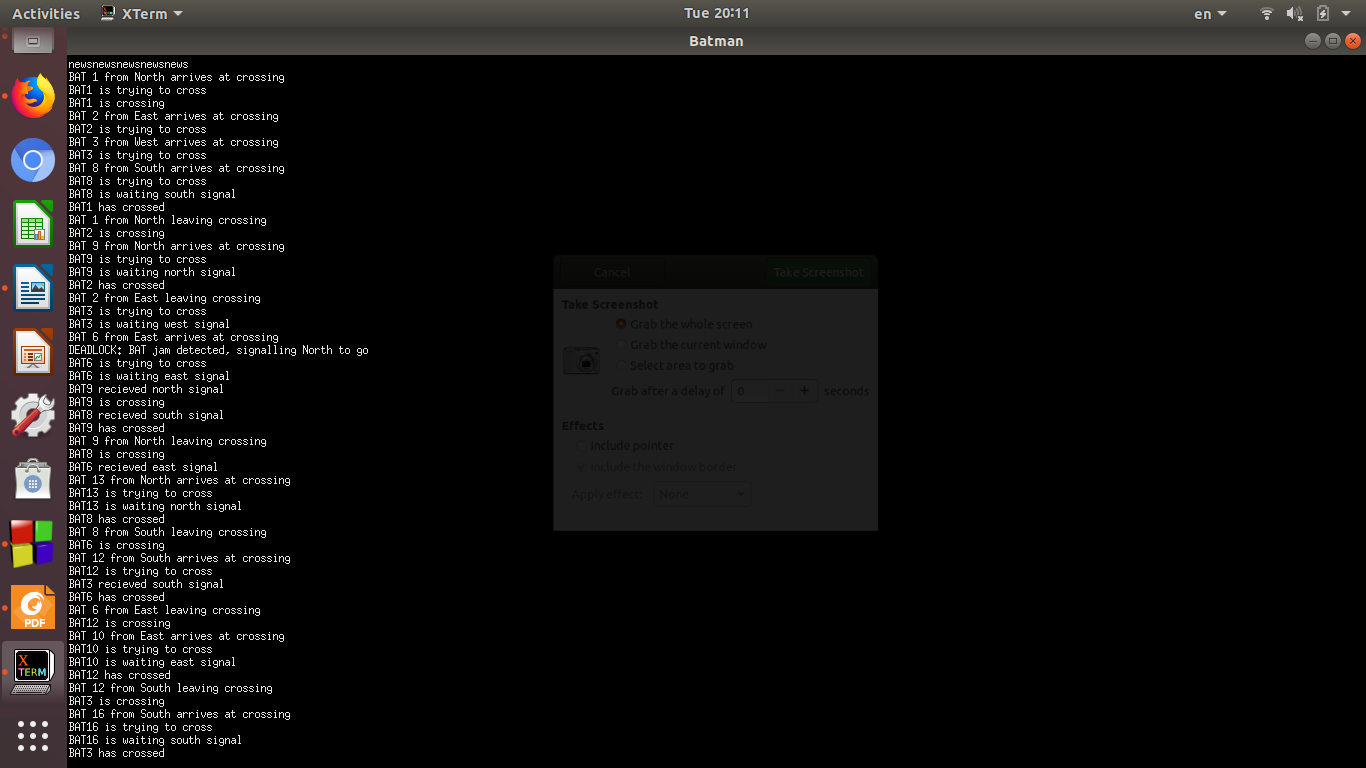
1-

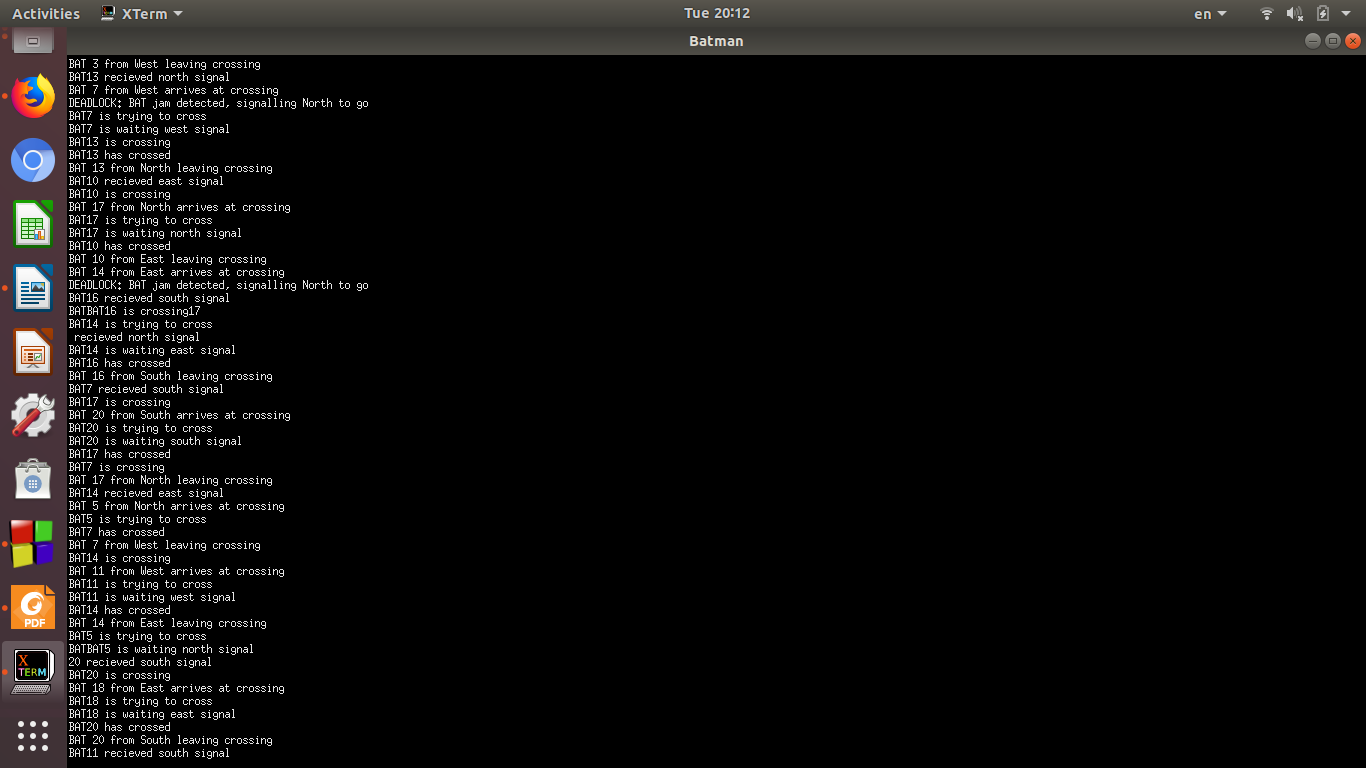
2-

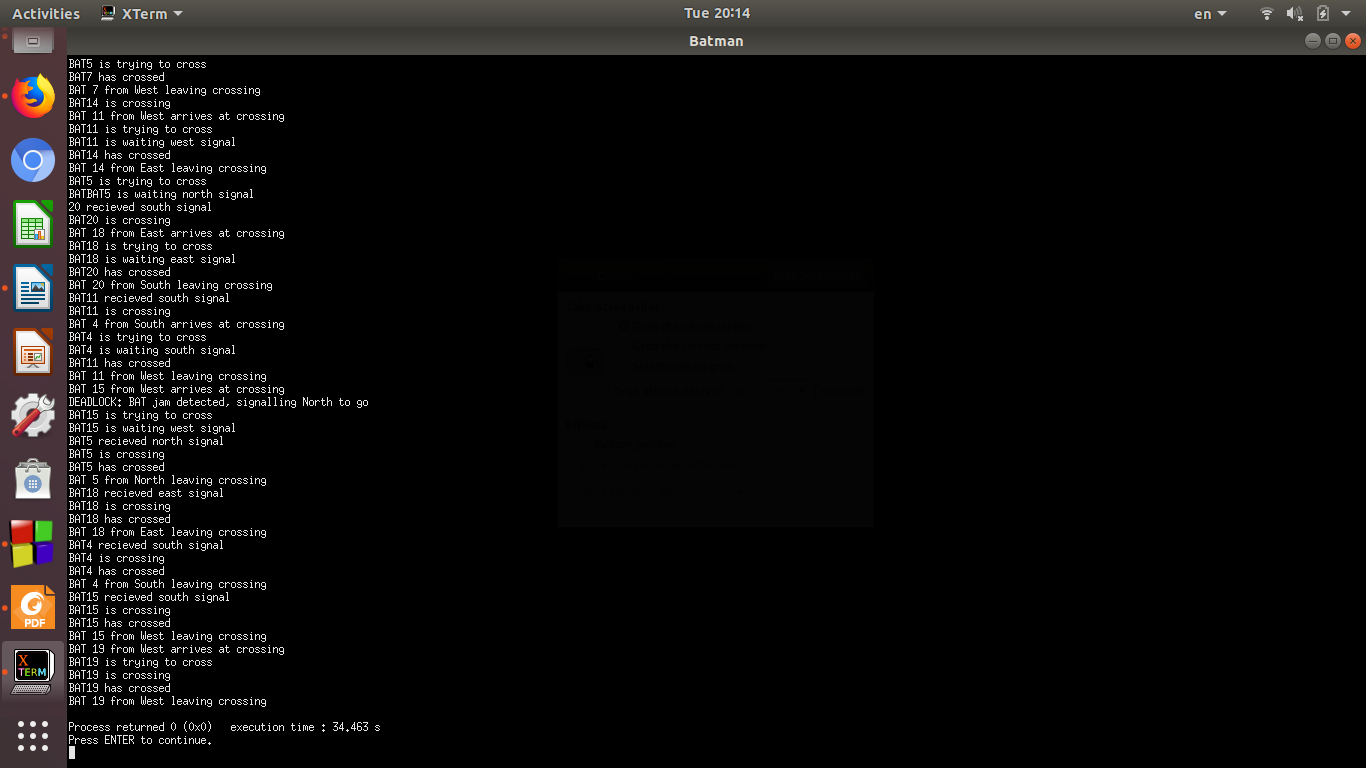
3-

****

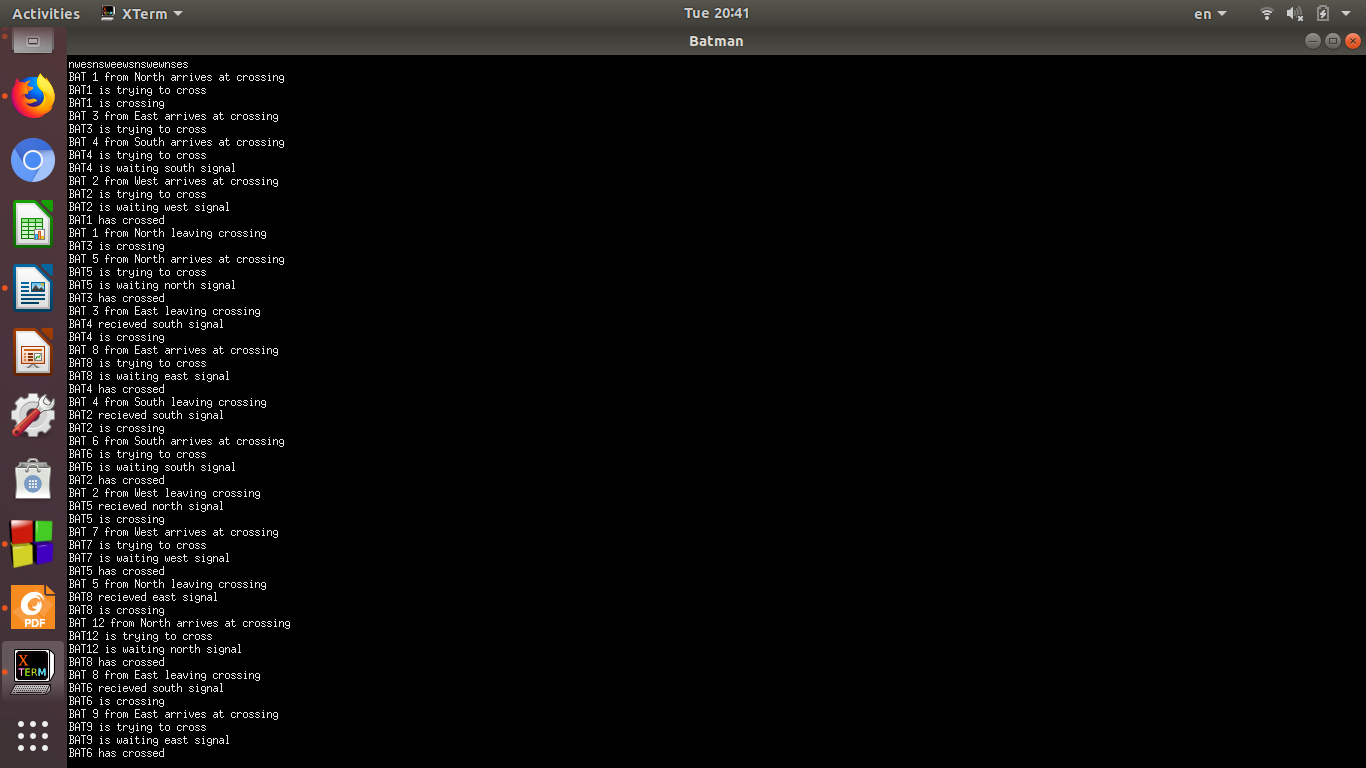
4-

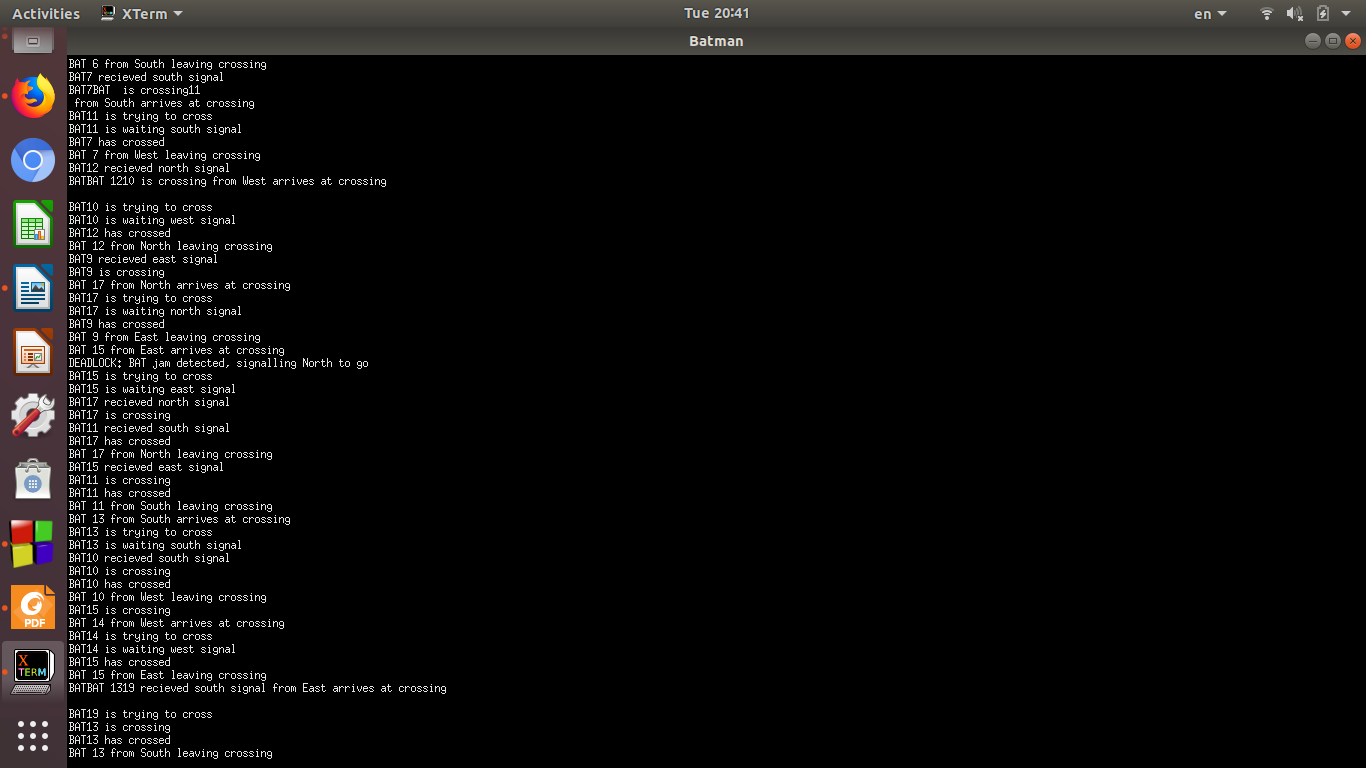
****

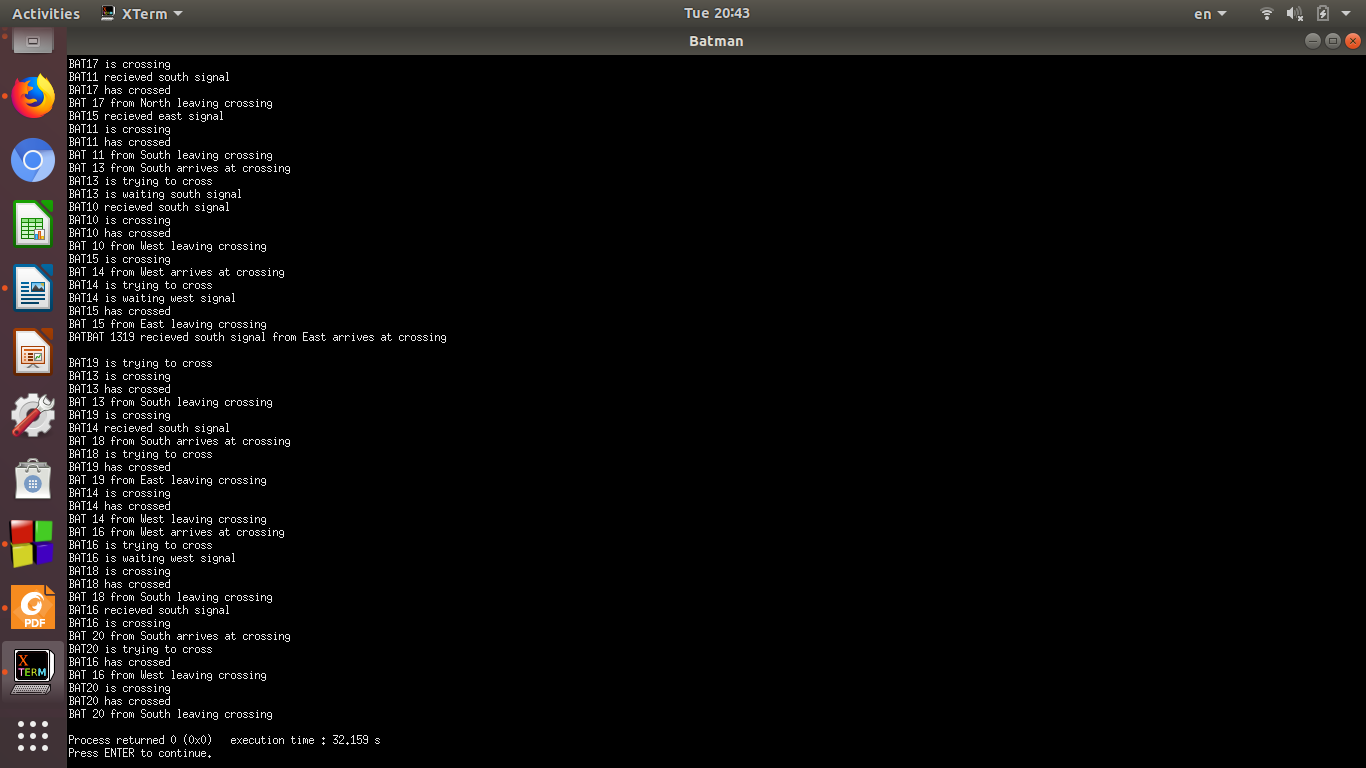
****

****

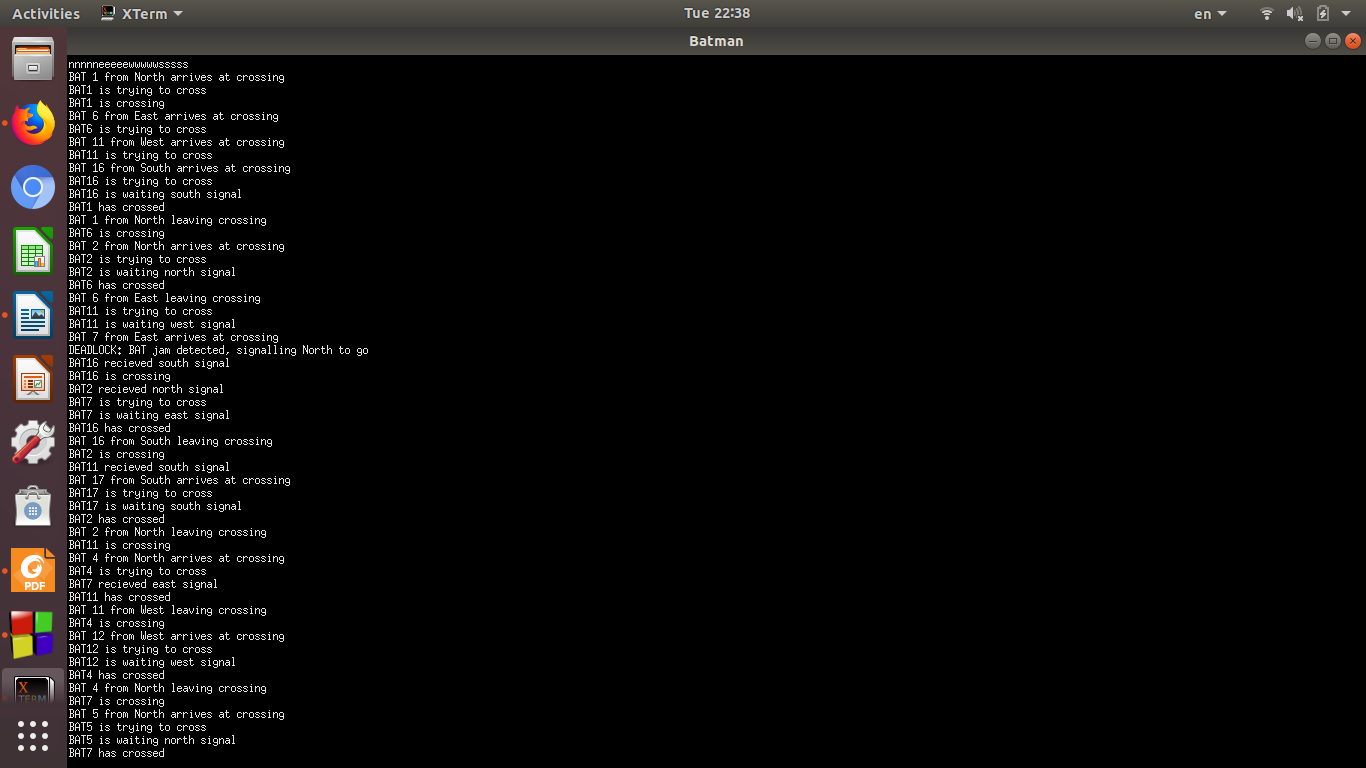
5-

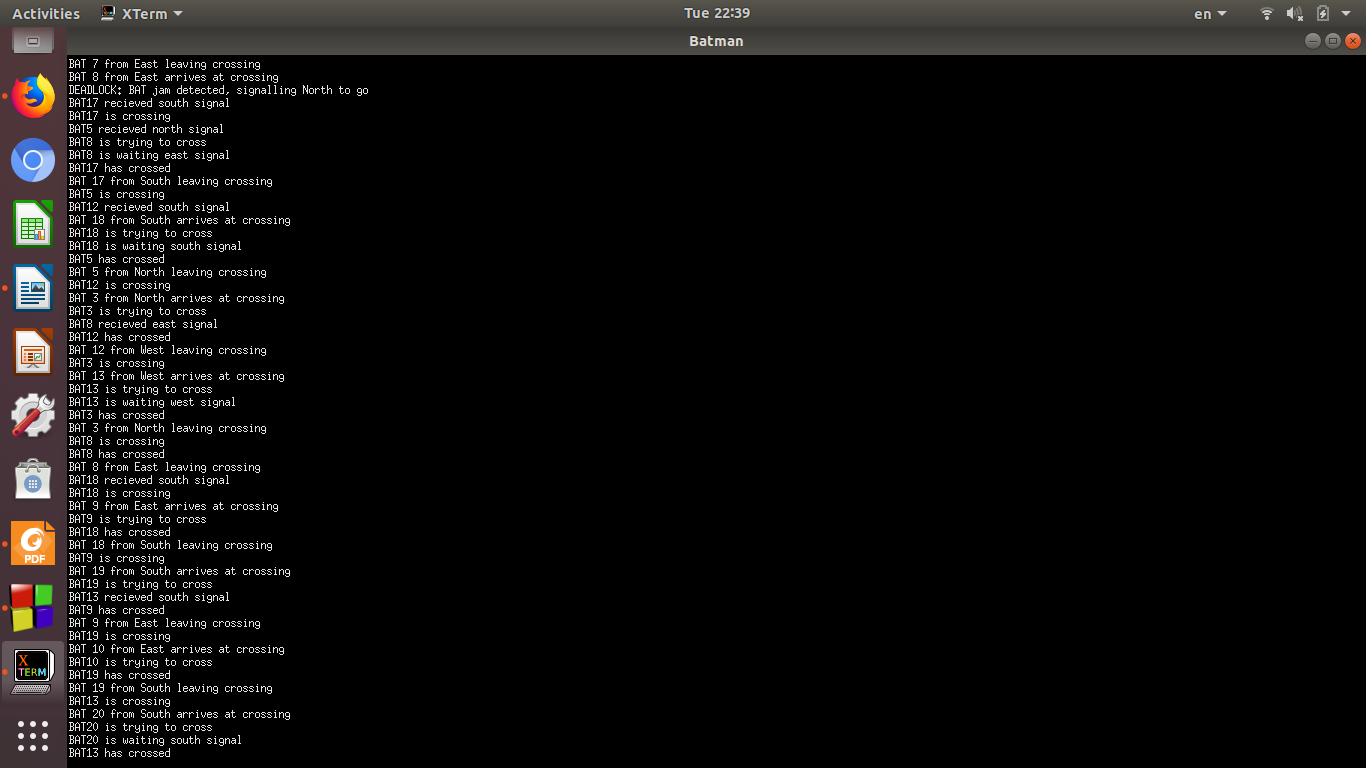


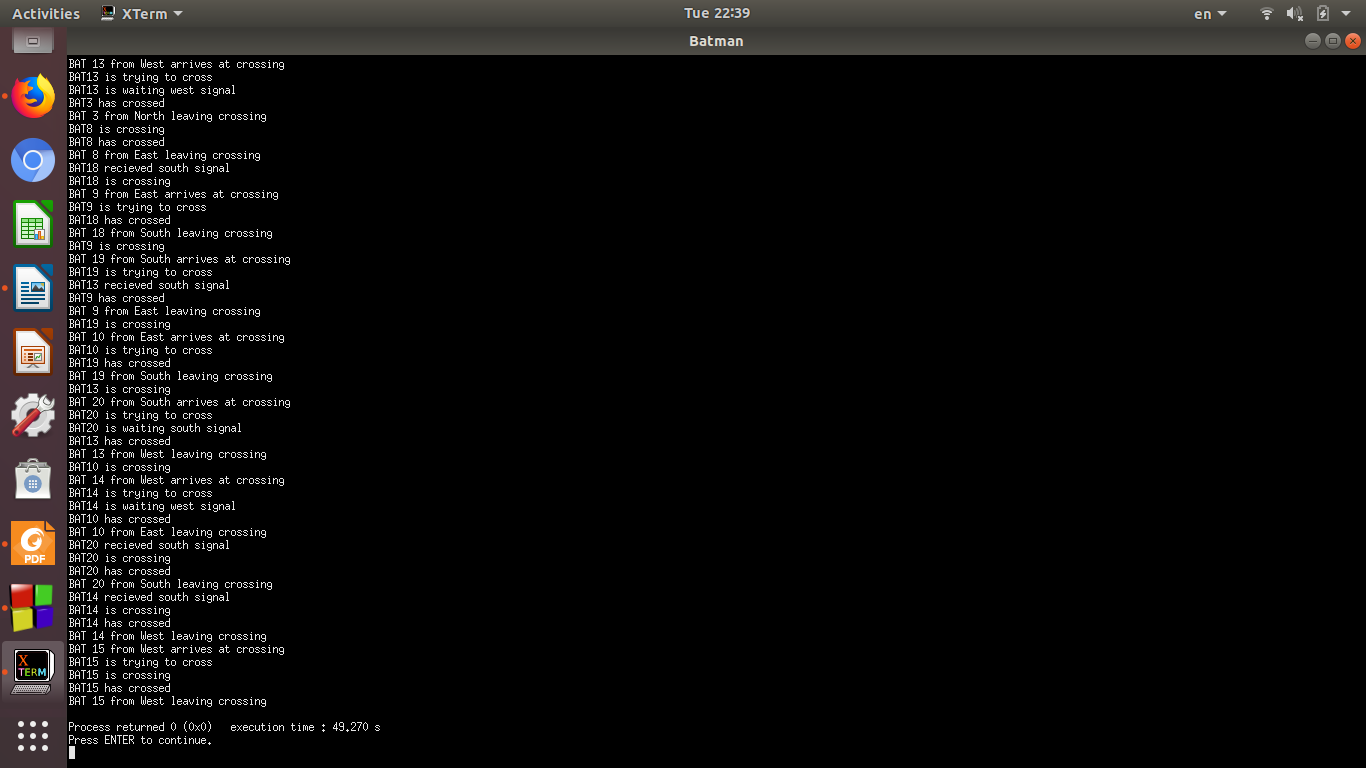


****

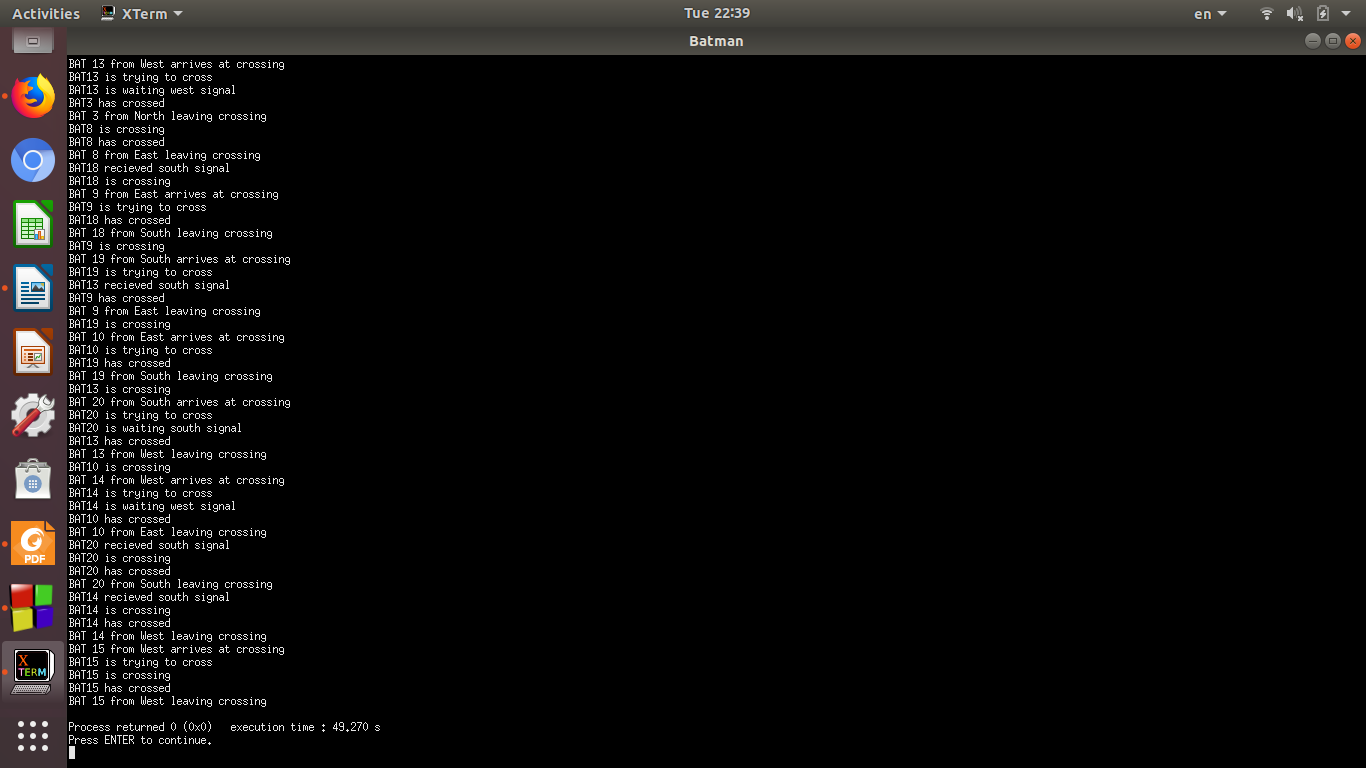
6-

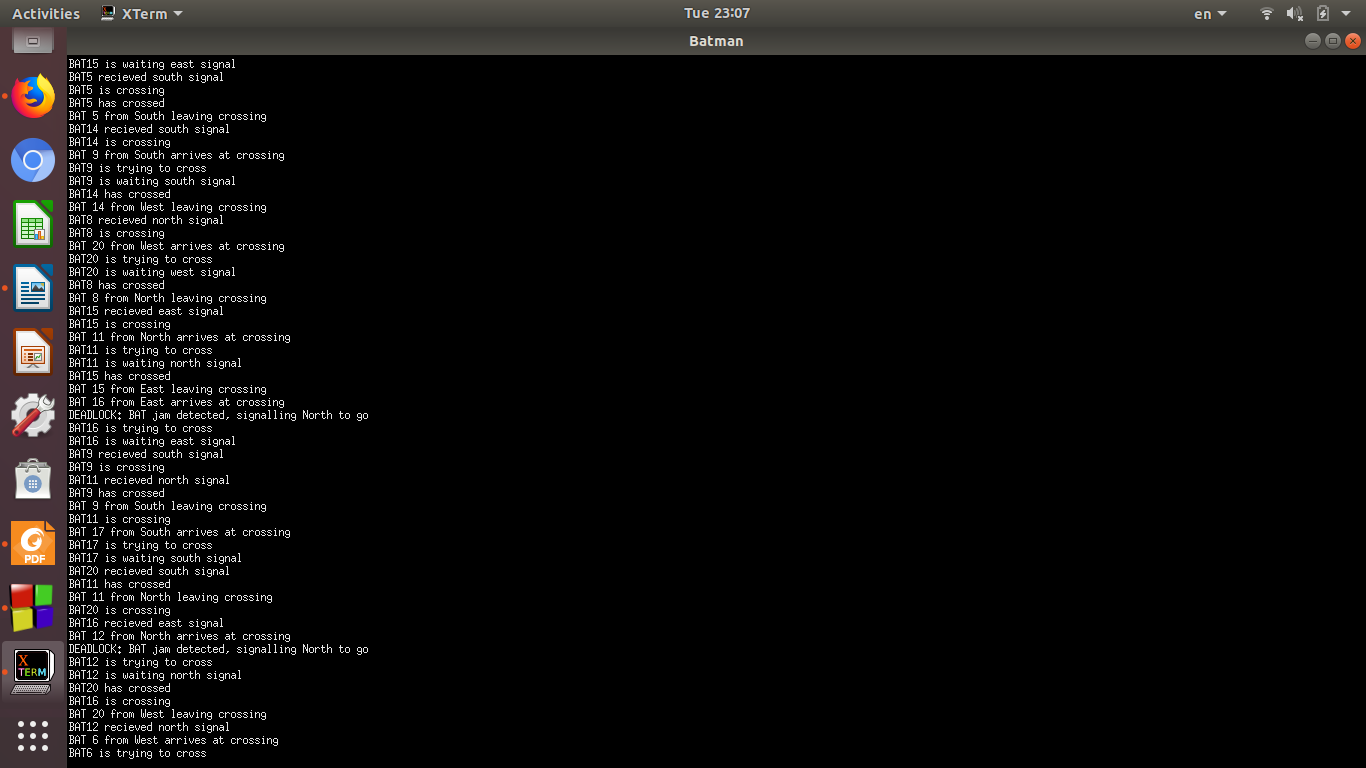


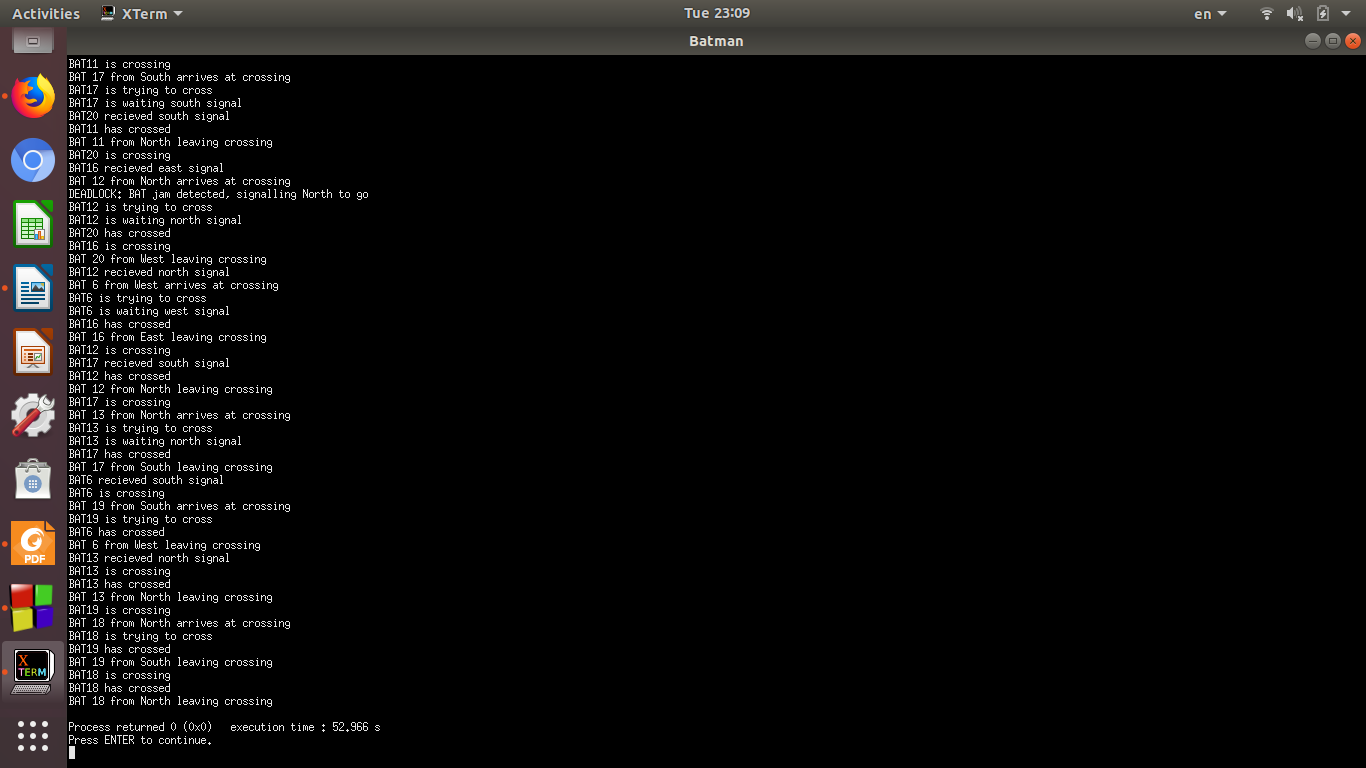




7-







“There is an interfering in a few printing lines in the screen shots which can be solved by applying a screen lock for all cout instructions (solved in code and from screen shot 6)”

“Check method is called after each arrive so in the steps the thread arrive then the deadlock is detected then the thread check it’s right and wait on signal and I think this happen due to unlock is slowing it a bit”

“The received signal is late a bit but it works right”

“A thread may receive a signal but another cross may happen first but this doesn’t lead to starvation as it will cross after that cross”

Name: Ahmed Mohamed Kamal EL-Bawab

ID: 09