Homework 2

Team 10
Apostolopoulou Ioanna
Toloudis Panagiotis

Assignment 1

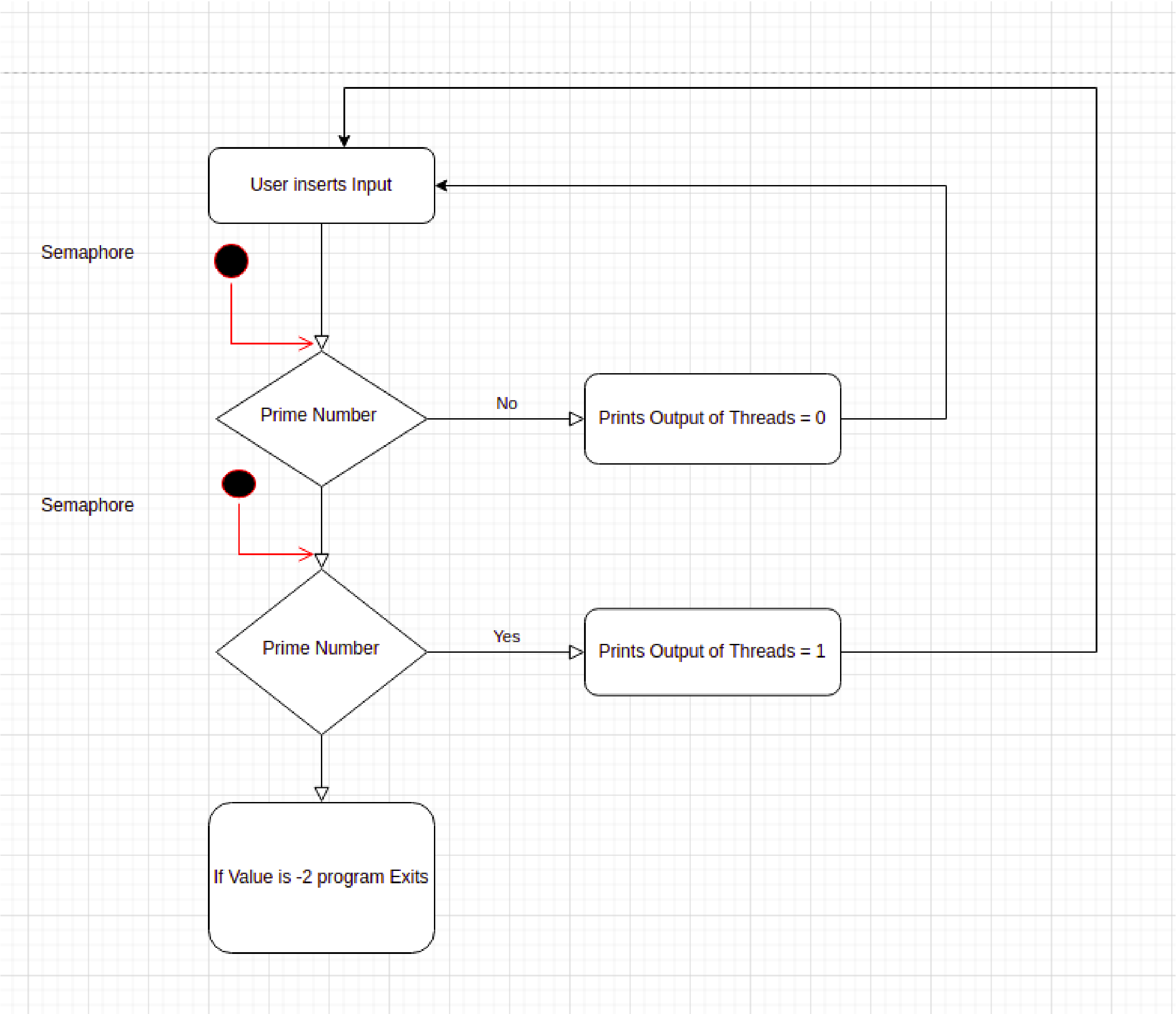
- **mysem_init()**
- Checks if n is valid
- Checks if Semaphore is already initialized if yes return 0
- Initializes the Semaphore by using systems v *semctl* function and returns 1

- **mysem_up()**
- Checks if Semaphore is initialized If not returns -1
- Checks if Semaphore is already Incremented and returns 0
- Makes the union Semaphore Up by using systems v semop
- Changes Semaphore operation in the Struct and returns 1

- **mysem_down()**
- Checks if Semaphore is initialized If not returns -1
- Makes the union Semaphore Down by using systems v semop
- Changes Semaphore operation in the Struct and returns 1

- **mysem_destroy()**
- Checks if Semaphore is initialized
- If it isn’t returns -1
- Destroy the Semaphore
- If Struct is Destroyed returns 1

Assignment 2


User inserts Input
Semaphore
Prime Number
No
Prints Output of Threads = 0
Semaphore
Prime Number
Yes
Prints Output of Threads = 1
If Value is -2 program Exits

Assignment 2

Program takes as Input random numbers from the test files and Creates Group Semaphores with size of the Thread and Initializes the Semaphore Values with 0.

When Threads exist and they are Available Semaphore Increment so a new Number is appointed to Thread and then the Decrement because Threads are not available.

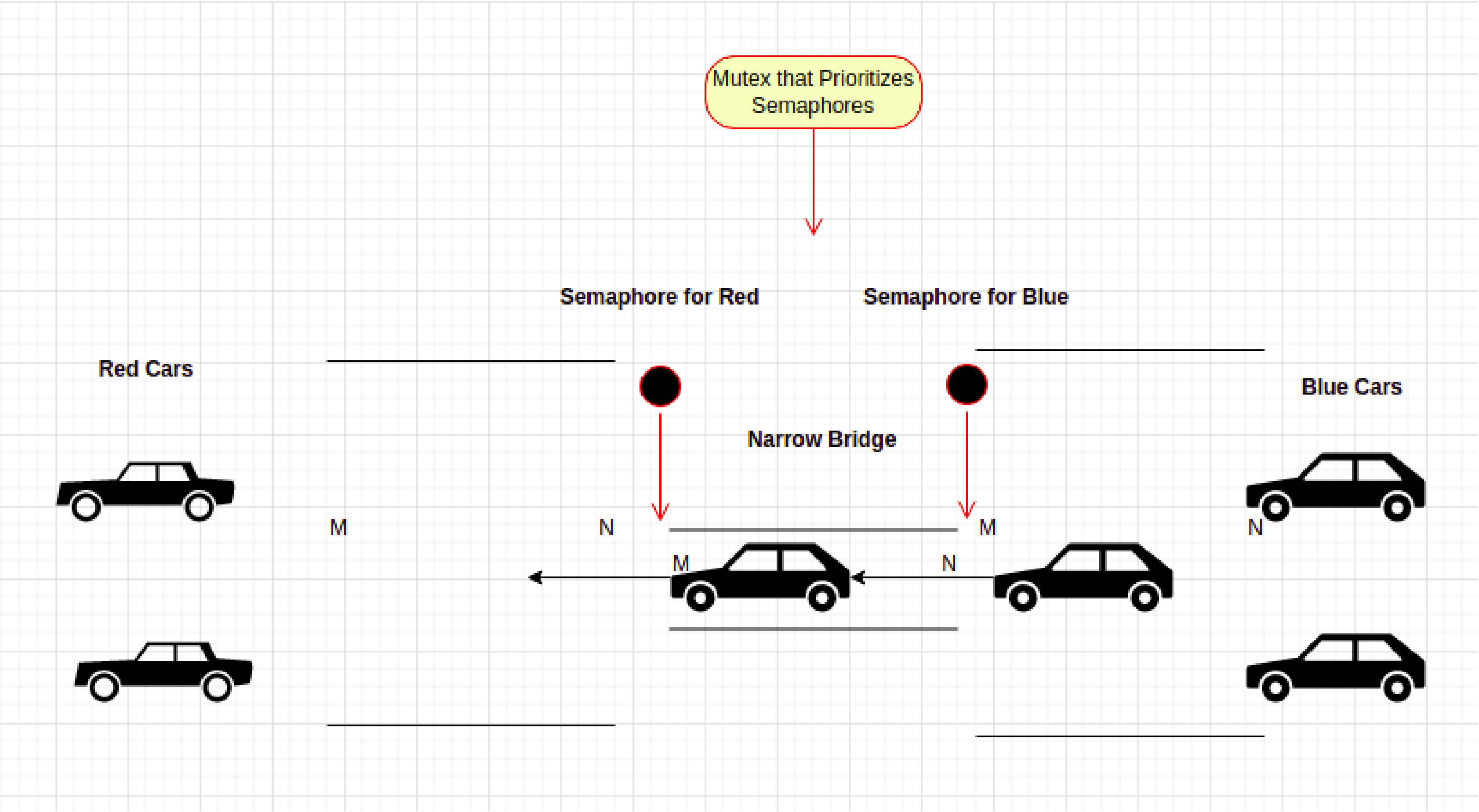
Program Retuns 0 if the number is not Prime.

And 1 if the number is Prime.

When in *Prime_Number() number is -1 Threads are notified that they should Exit.

When Value in Main becomes -2 the program Exits and Destroy the Semaphores and Threads.

Assignment 3



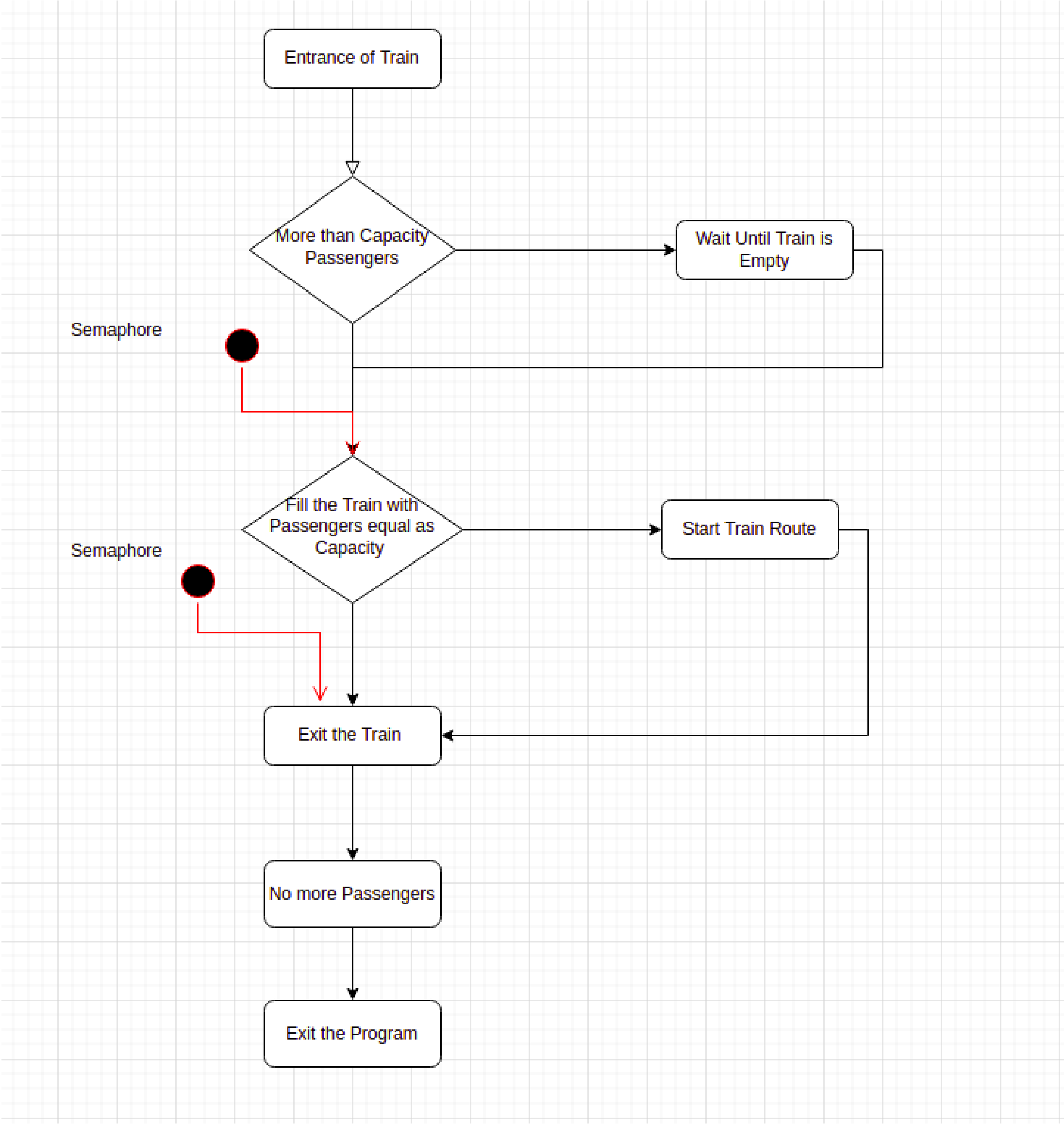
Assignment 3

- Arriving_Cars()
- Checks the color of the Car that Arrived at the edge of the Bridge
- Checks if Bridge reached Full Capacity
- If specific color has priority
- And if there are not Cars of other Color Waiting
- If all above are true Car should Cross the Bridge

- Leaving_Cars()
- Checks if the Car_On_Bridge is the last on Bridge
- Checks the color of Cars on the Bridge
- Checks if there is only one Car on the Bridge
- Prioritizes the turn of Color of Cars
- Increments the Semaphore so Cars Continue with the right turn

- Main()
- Create and Initialize the Struct of the Bridge and the Semaphores
- Reads from file the amount of Cars, the Color and the time they need to generate new ones
- If the amount of Cars given is negative program Exits
- Destroy the Semaphores and Free all allocated Memory

Assignment 4


Entrance of Train
More than Capacity Passengers
Wait Until Train is Empty
Semaphore
Fill the Train with Passengers equal as Capacity
Start Train Route
Semaphore
Exit the Train
No more Passengers
Exit the Program

Assignment 4

- fill_train()
- Checks if Train reached max Capacity of Passengers
- If the Train is full waits until it Empties
- If the exit flag equals to 1 checks if Passengers Waiting are less than max Capacity of the Train
- If Passengers are more than max Capacity of the Train and Train is Empty exactly the amount of the Capacity boards on the Train and the rest are Waiting
- Else the Train stops its Routes and Passengers boarded Get Off the Train

- Start_Train()
- Creates the Train Thread
- Waits for the next Group of Passengers to Arrive
- Wait until Train has reached max Capacity of Passengers
- Starts the Train Route
- Informs the Passengers to Get Off the Train because Route is Finished
- If no Passengers are on the Train and Exit flag equals to 1 Train stops every Route

- Main()
- Creates Group of Semaphores and Initializes the Semaphore Values to 0
- Creates the Train Thread
- Reads the input from the files
- Creates Passenger Thread
- Waits for All Passengers to make Train Route and Get off the Train
- Destroy Semaphores and Free all Allocated Memory
- Exits the Program