

1.b La estructura de la solución consta de 5 objetos principales: La clase Automóvil, la clase Puente, la Clase Sentido, los Hilos en java y el Semáforo para evitar en el problema de un choque de carros.

Quienes piden los recursos en este caso para poder pasar el puente son los Automóviles que se dividen en clase derecha e izquierda. La Clase Sentido también pide recursos y este es el encargado de cierto tiempo cambiar el sentido en que se pueden transportar los carros del puente.

Quien devuelve los recursos o los libera para cambiar el sentido en que puedan pasar los carros en este caso sería el mismo Puente.

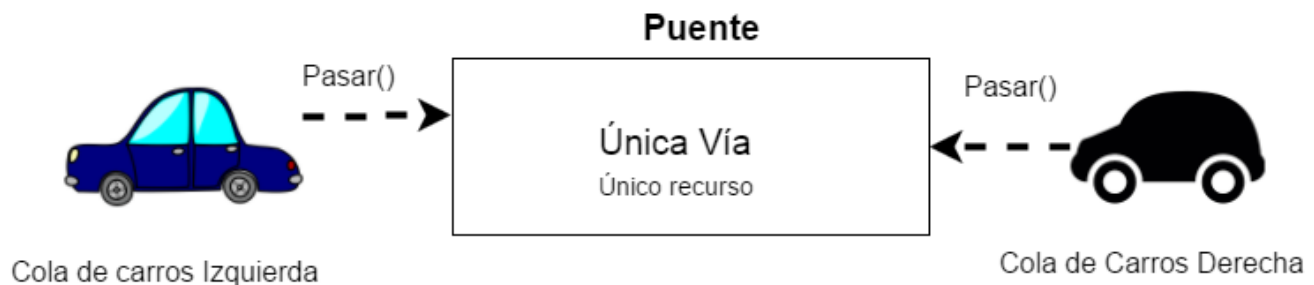
La clase automóvil constará de dos tipos: TipoIzquierda y TipoDerecha, esto nos indicará en qué lado del puente está. Cada vez que lleguen carros de cualquier tipo se añadirán a la cola del tipo correspondiente en la clase Puente.

La clase Sentido tiene la responsabilidad de ser un semáforo en vida real (Distinto a los semáforos de java), es decir que cada cierto tiempo da paso a que los carros del sentido derecho puedan pasar sobre el puente o los del sentido izquierdo puedan pasar el puente, esta es la clase que evitará el problema de la inanición dando paso a los carros de la derecha o izquierda.

La Clase puente es aquella quien tiene la cantidad de carros que están en sus dos lados, así como es el encargado de contar que la cantidad de carros que estén sobre él no sobrepasen a 10 automóviles.

De esta manera con los hilos y los Semáforos de Java para controlar el pedido de recursos (pasar por el puente) simularemos un ambiente real del sistema.

Explicaremos la solución:



Como muestra la gráfica existe el problema de que el puente sólo posee una vía lo que quiere decir que sólo puede dar paso a carros de un sentido a otro pero no los dos al mismo tiempo. Por eso creé la clase Sentido que se encarga de darle paso a los dos lados del puente cada cierto tiempo variable.

Solucionado el problema de dar paso cada cierto tiempo a sólo alguno de los dos lados empezamos entonces a resolver los problemas de sincronización de datos. ¿Qué pasa si cambia el sentido de poder pasar los carros, pero aún sobre el puente existen carros tratando de llegar a su otro lado? ¿Cuándo están pasando los carros de un sentido y llega un carro nuevo para agregarse a la cola? ¿Si llega un carro y el sentido de ese lado está en verde para poder pasar, pero existe una cola de carros delante de

¿l? Si no se trata de manejar los recursos para mantener los datos íntegros el programa colapsa con resultados que no simulan un comportamiento en vida real

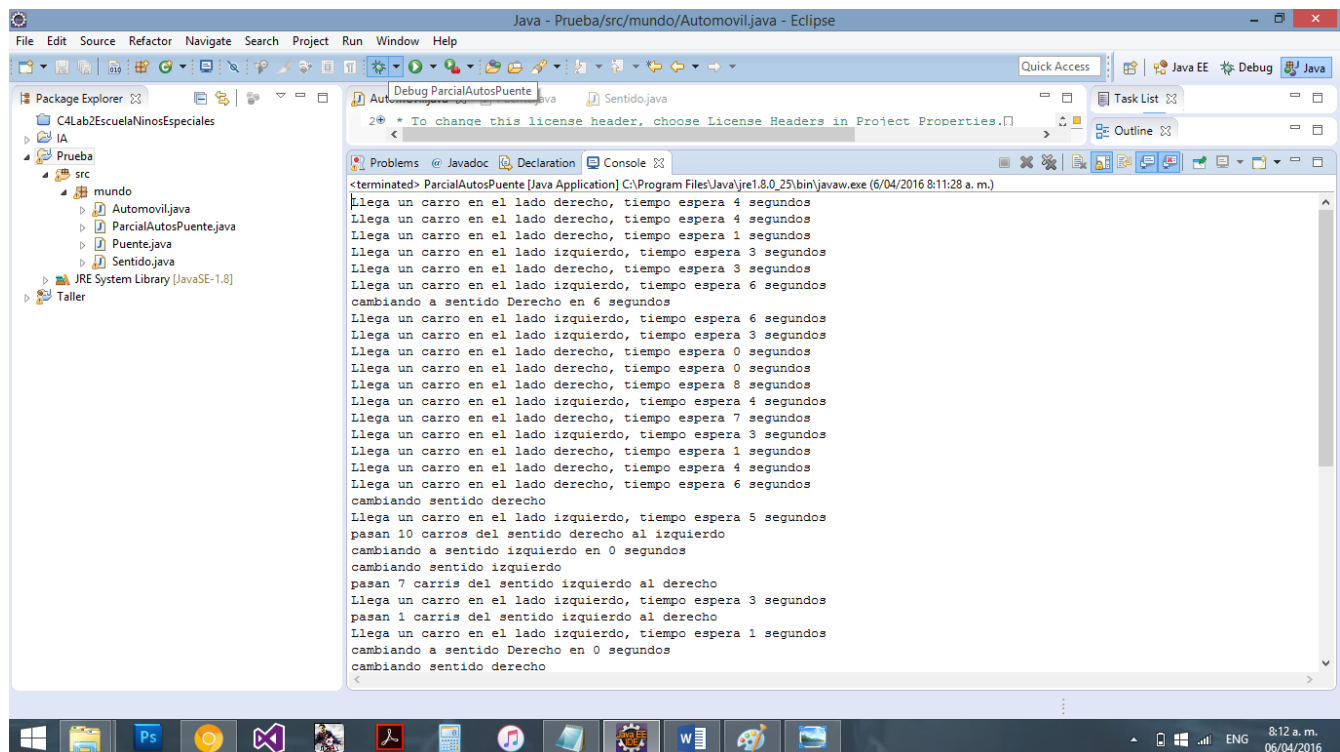
Por eso se crean 3 Semáforos de Java:

2 son los encargados de asignar los recursos para aumentar la cola de carros en cada lado.

El próximo semáforo está acompañado de una validación por parte de la clase Sentido cambiando en un tiempo indeterminado una variable sentido que posee dos valores: Izquierda y Derecha.

Cuando se valide en qué sentido es que pueden pasar los carros se procede a llamar el método que permite el paso de los carros, pero para mantener la sincronización de procesos de los hilos que llaman al mismo tiempo este método se crea un semáforo que asigna un solo recurso para atender un proceso a la vez y disminuir la cantidad de carros en espera que se da.

Este es un ejemplo de lo que arroja el programa:



```
<terminated> ParcialAutosPunte [Java Application] C:\Program Files\Java\jre1.8.0_25\bin\javaw.exe (6/04/2016 8:11:28 a.m.)
Llega un carro en el lado derecho, tiempo espera 4 segundos
Llega un carro en el lado derecho, tiempo espera 4 segundos
Llega un carro en el lado derecho, tiempo espera 1 segundos
Llega un carro en el lado izquierdo, tiempo espera 3 segundos
Llega un carro en el lado derecho, tiempo espera 3 segundos
Llega un carro en el lado izquierdo, tiempo espera 6 segundos
cambiando a sentido Derecho en 6 segundos
Llega un carro en el lado izquierdo, tiempo espera 6 segundos
Llega un carro en el lado izquierdo, tiempo espera 3 segundos
Llega un carro en el lado derecho, tiempo espera 0 segundos
Llega un carro en el lado derecho, tiempo espera 0 segundos
Llega un carro en el lado derecho, tiempo espera 8 segundos
Llega un carro en el lado izquierdo, tiempo espera 4 segundos
Llega un carro en el lado derecho, tiempo espera 7 segundos
Llega un carro en el lado izquierdo, tiempo espera 3 segundos
Llega un carro en el lado derecho, tiempo espera 1 segundos
Llega un carro en el lado derecho, tiempo espera 4 segundos
Llega un carro en el lado derecho, tiempo espera 6 segundos
cambiando sentido derecho
Llega un carro en el lado izquierdo, tiempo espera 5 segundos
pasan 10 carros del sentido derecho al izquierdo
cambiando a sentido izquierdo en 0 segundos
cambiando sentido izquierdo
pasan 7 carros del sentido izquierdo al derecho
Llega un carro en el lado izquierdo, tiempo espera 3 segundos
pasan 1 carros del sentido izquierdo al derecho
Llega un carro en el lado izquierdo, tiempo espera 1 segundos
cambiando a sentido Derecho en 0 segundos
cambiando sentido derecho
```

Java - Prueba/src/mundo/Automovil.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access Java EE Debug Java

Package Explorer

- C4Lab2EscuelaNinosEspeciales
 - IA
 - Prueba
 - mundo
 - Automovil.java
 - ParcialAutosPunte.java
 - Punte.java
 - Sentido.java
 - JRE System Library [JavaSE-1.8]
 - Taller

Debug PartialAutosPunte.java Sentido.java

2 * To change this license header, choose License Headers in Project Properties.

Problems Javadoc Declaration Console

<terminated> ParcialAutosPunte [Java Application] C:\Program Files\Java\jre1.8.0_25\bin\javaw.exe (6/04/2016 8:11:28 a. m.)

```
Llega un carro en el lado derecho, tiempo espera 4 segundos
Llega un carro en el lado derecho, tiempo espera 4 segundos
Llega un carro en el lado derecho, tiempo espera 1 segundos
Llega un carro en el lado izquierdo, tiempo espera 3 segundos
Llega un carro en el lado derecho, tiempo espera 3 segundos
Llega un carro en el lado izquierdo, tiempo espera 6 segundos
cambiando a sentido Derecho en 6 segundos
Llega un carro en el lado izquierdo, tiempo espera 6 segundos
Llega un carro en el lado izquierdo, tiempo espera 3 segundos
Llega un carro en el lado derecho, tiempo espera 0 segundos
Llega un carro en el lado derecho, tiempo espera 0 segundos
Llega un carro en el lado derecho, tiempo espera 8 segundos
Llega un carro en el lado izquierdo, tiempo espera 4 segundos
Llega un carro en el lado derecho, tiempo espera 7 segundos
Llega un carro en el lado izquierdo, tiempo espera 3 segundos
Llega un carro en el lado derecho, tiempo espera 1 segundos
Llega un carro en el lado derecho, tiempo espera 4 segundos
Llega un carro en el lado derecho, tiempo espera 6 segundos
cambiando sentido derecho
Llega un carro en el lado izquierdo, tiempo espera 5 segundos
pasan 10 carros del sentido derecho al izquierdo
cambiando a sentido izquierdo en 0 segundos
cambiando sentido izquierdo
pasan 7 carris del sentido izquierdo al derecho
Llega un carro en el lado izquierdo, tiempo espera 3 segundos
pasan 1 carris del sentido izquierdo al derecho
Llega un carro en el lado izquierdo, tiempo espera 1 segundos
cambiando a sentido Derecho en 0 segundos
cambiando sentido derecho
```

Sleep del thread

Notificacion del tiempo en que se cambiara el sentido

Da paso a los carros que estan en el lado derecho, maximo pueden pasar 10

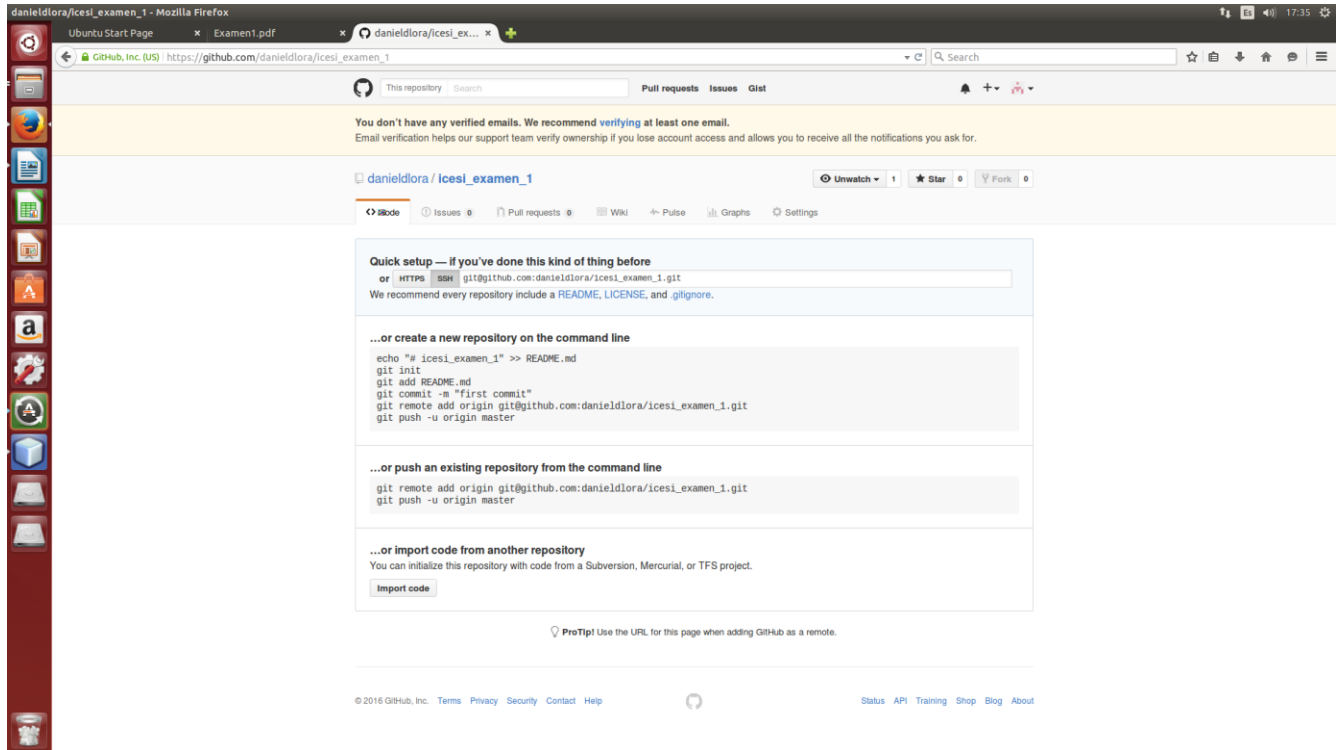
Si llega un carro y el semaforo de su lado (Clase sentido) esta en verde, puede proceder a pasar directamente

Windows Taskbar: 8:12 a. m. 06/04/2016

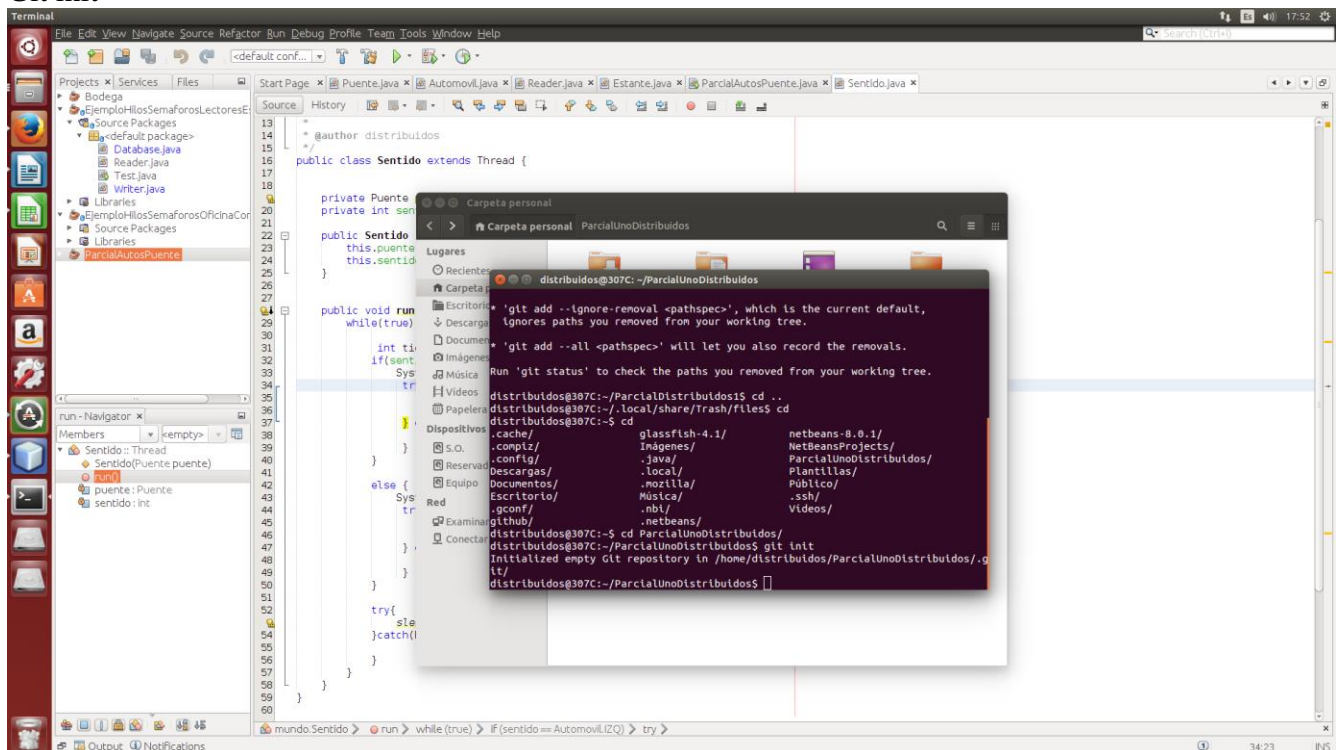
2.

Proceso de subida de GIT

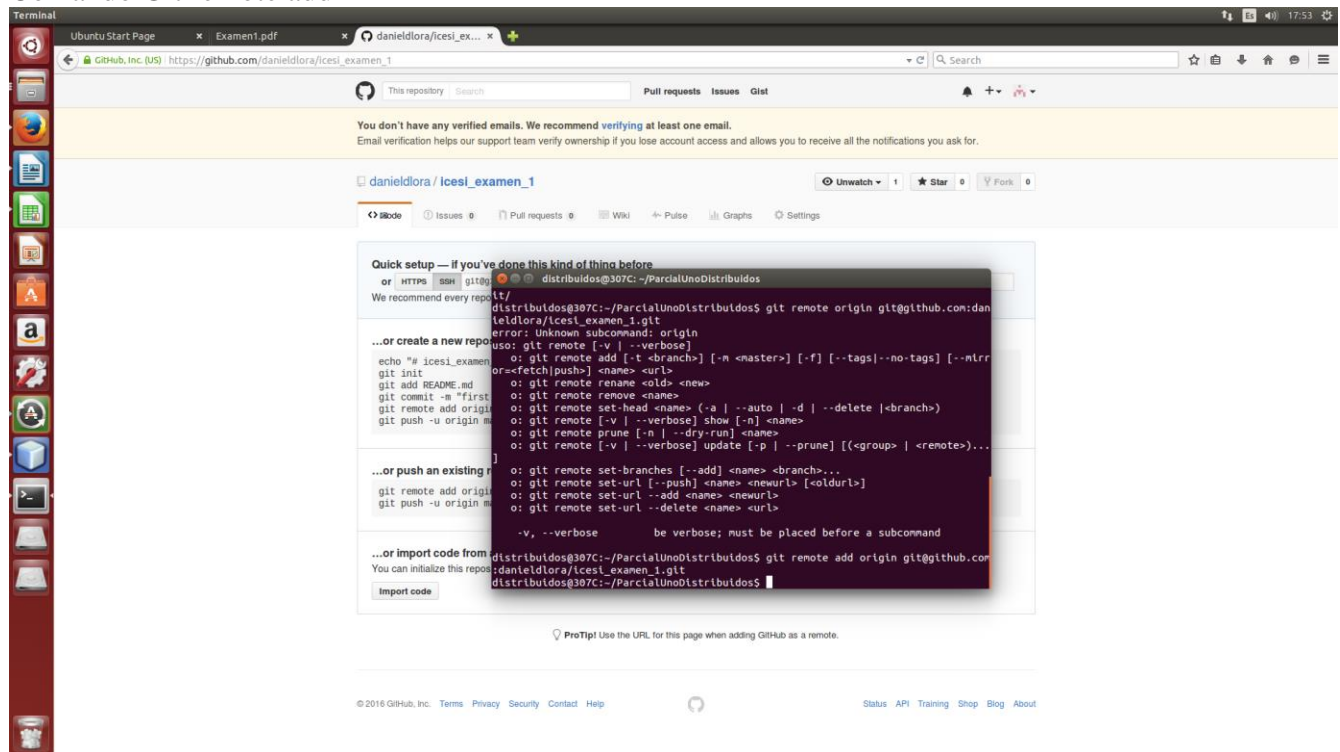
El proceso de subida de Github se hizo por SSH



Git init



Comando Git remote add

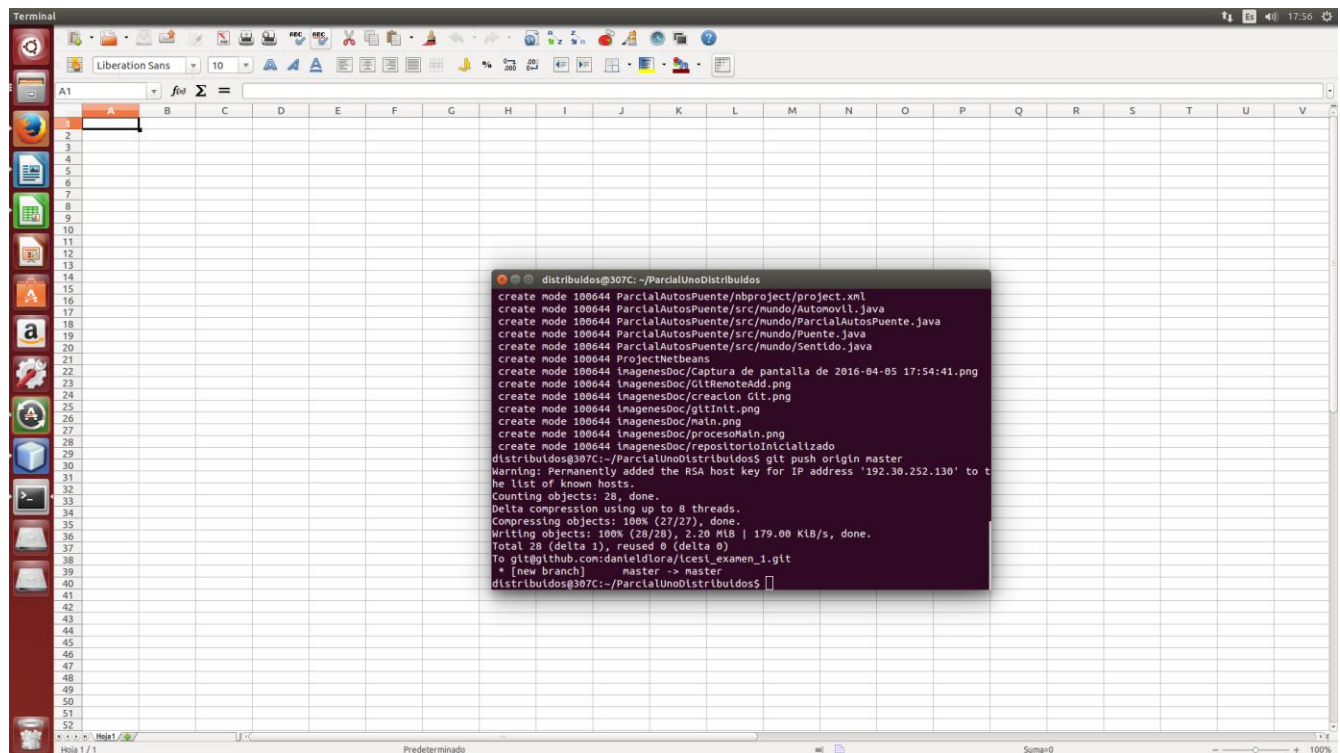


The screenshot shows a web browser displaying the GitHub repository page for 'danielldora/icesi_examen_1'. The page includes a message about email verification and a 'Quick setup' section. A terminal window is overlaid on the page, showing the following commands and output:

```
distribuidos@307C:~/ParcialUnoDistribuidos$ git remote origin git@github.com:danielldora/icesi_examen_1.git
error: Unknown subcommand: origin
usage: git remote [-v] [--verbose]
       o: git remote add [-t <branch>] [-m <master>] [-f] [--tags|--no-tags] [--mirror]
       o: git remote add [-t <branch>] [-m <master>] [-f] [--tags|--no-tags] [--mirror]
       o: git remote rename <old> <new>
       o: git remote remove <name>
       o: git remote set-head <name> (-a | --auto | -d | --delete | <branch>)
       o: git remote [-v] [--verbose] show [-n] <name>
       o: git remote prune [-n | --dry-run] <name>
       o: git remote [-v] [--verbose] update [-p | --prune] [(<group> | <remote>)...]
       o: git remote set-branches [--add] <name> <branch>...
       o: git remote set-url [--push] <name> <newurl> [<oldurl>]
       o: git remote set-url --add <name> <newurl>
       o: git remote set-url --delete <name> <url>

-v, --verbose          be verbose; must be placed before a subcommand

distribuidos@307C:~/ParcialUnoDistribuidos$ git remote add origin git@github.com:danielldora/icesi_examen_1.git
distribuidos@307C:~/ParcialUnoDistribuidos$
```

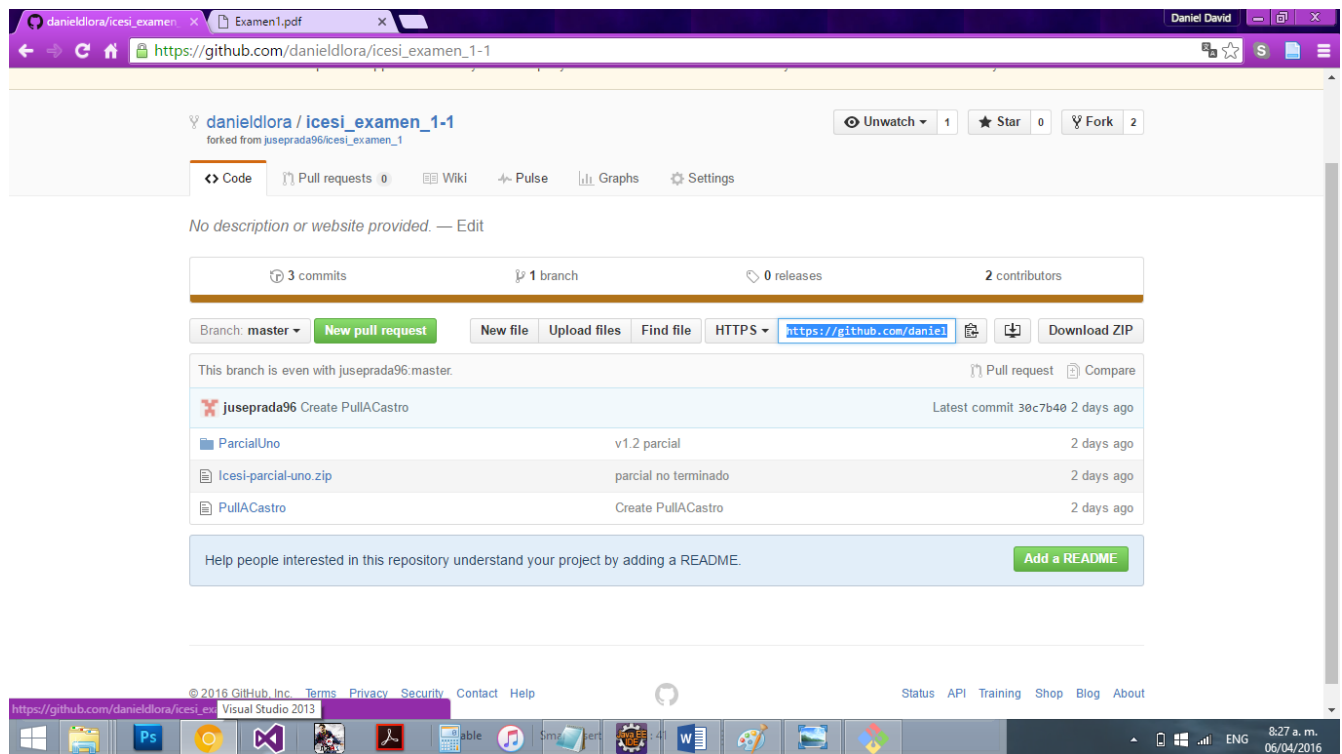


The screenshot shows a terminal window overlaid on a spreadsheet application. The terminal output shows the result of a 'git push' command:

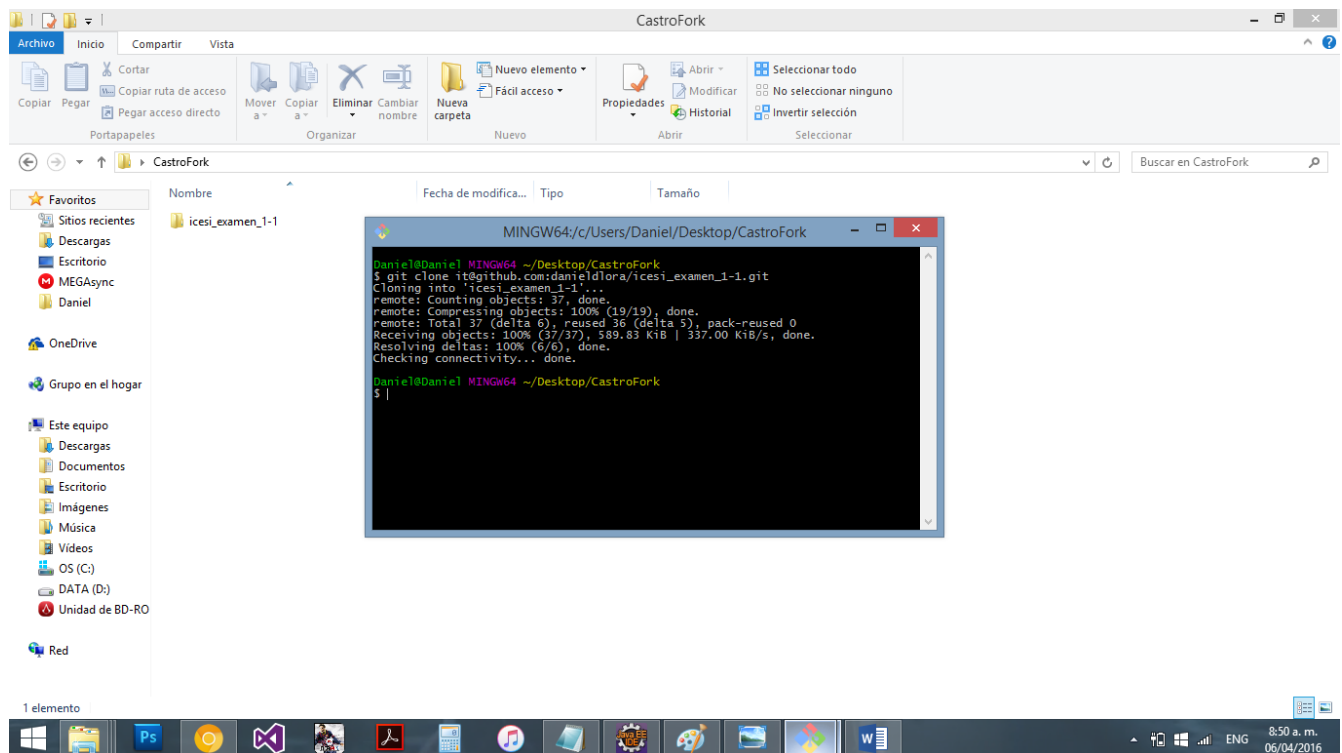
```
distribuidos@307C:~/ParcialUnoDistribuidos$ git push origin master
create mode 100644 ParcialAutosPuede/nbproject/project.xml
create mode 100644 ParcialAutosPuede/src/mundo/Automov1.java
create mode 100644 ParcialAutosPuede/src/mundo/ParcialAutosPuede.java
create mode 100644 ParcialAutosPuede/src/mundo/Puente.java
create mode 100644 ParcialAutosPuede/src/mundo/Sentido.java
create mode 100644 ProjectWebBeans
create mode 100644 InagenesDoc/Captura de pantalla de 2016-04-05 17:54:41.png
create mode 100644 InagenesDoc/gitRemoteadd.png
create mode 100644 InagenesDoc/creacion Git.png
create mode 100644 InagenesDoc/gitInit.png
create mode 100644 InagenesDoc/main.png
create mode 100644 InagenesDoc/procesoMain.png
create mode 100644 InagenesDoc/repositorioInicializado
distribuidos@307C:~/ParcialUnoDistribuidos$ git push origin master
Warning: Permanently added the RSA host key for IP address '192.30.252.130' to the list of known hosts.
Counting objects: 28, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (27/27), done.
Writing objects: 100% (28/28), 2.20 MiB | 179.00 KiB/s, done.
Total 28 (delta 1), reused 0 (delta 0)
To git@github.com:danielldora/icesi_examen_1.git
 * [new branch]      master -> master
distribuidos@307C:~/ParcialUnoDistribuidos$
```

Link Respositorio: https://github.com/danielldora/icesi_examen_1

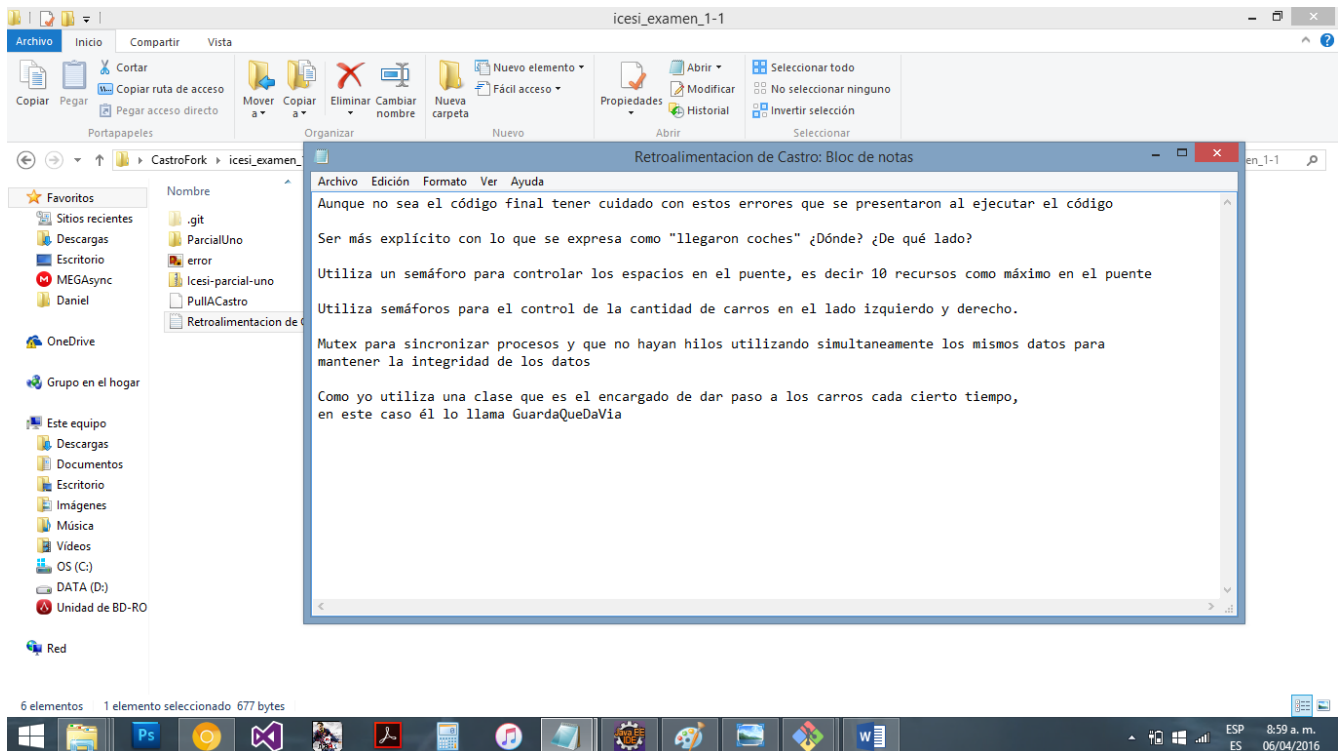
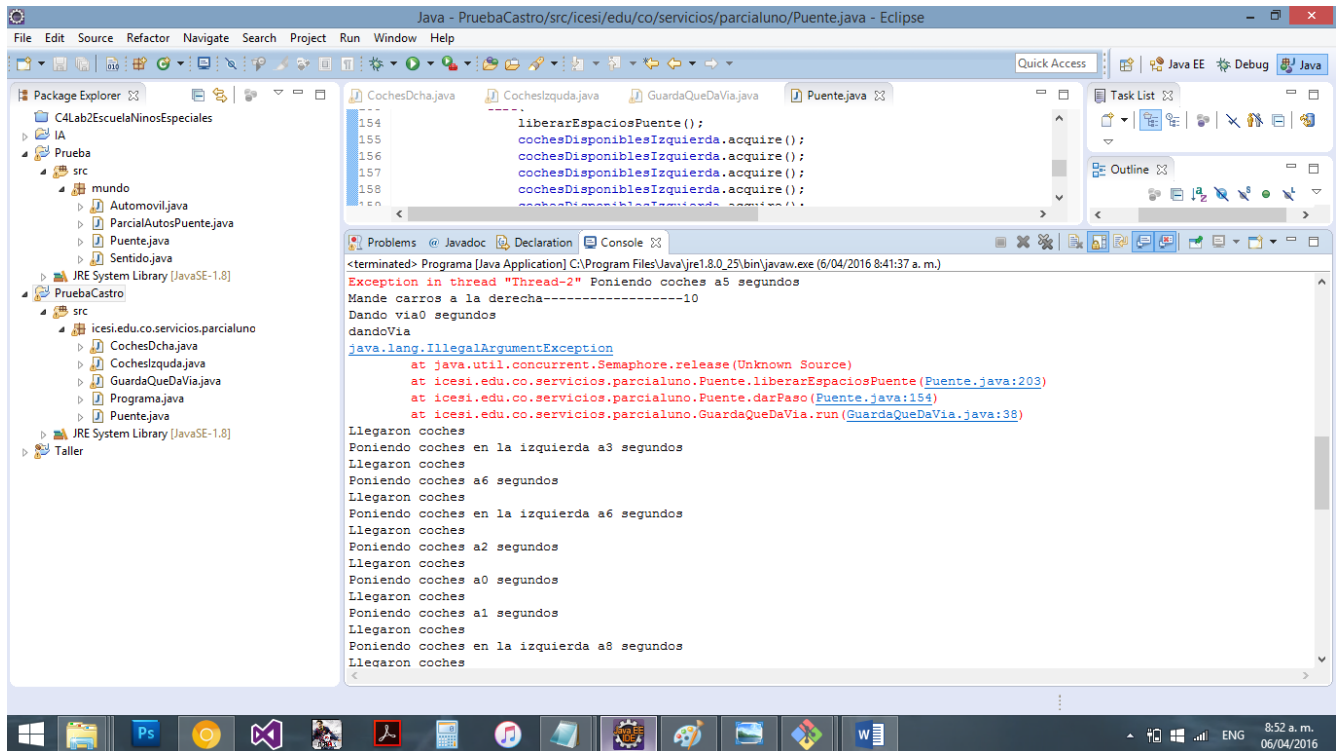
3. Fork al repositorio del estudiante Juan Pablo Castro



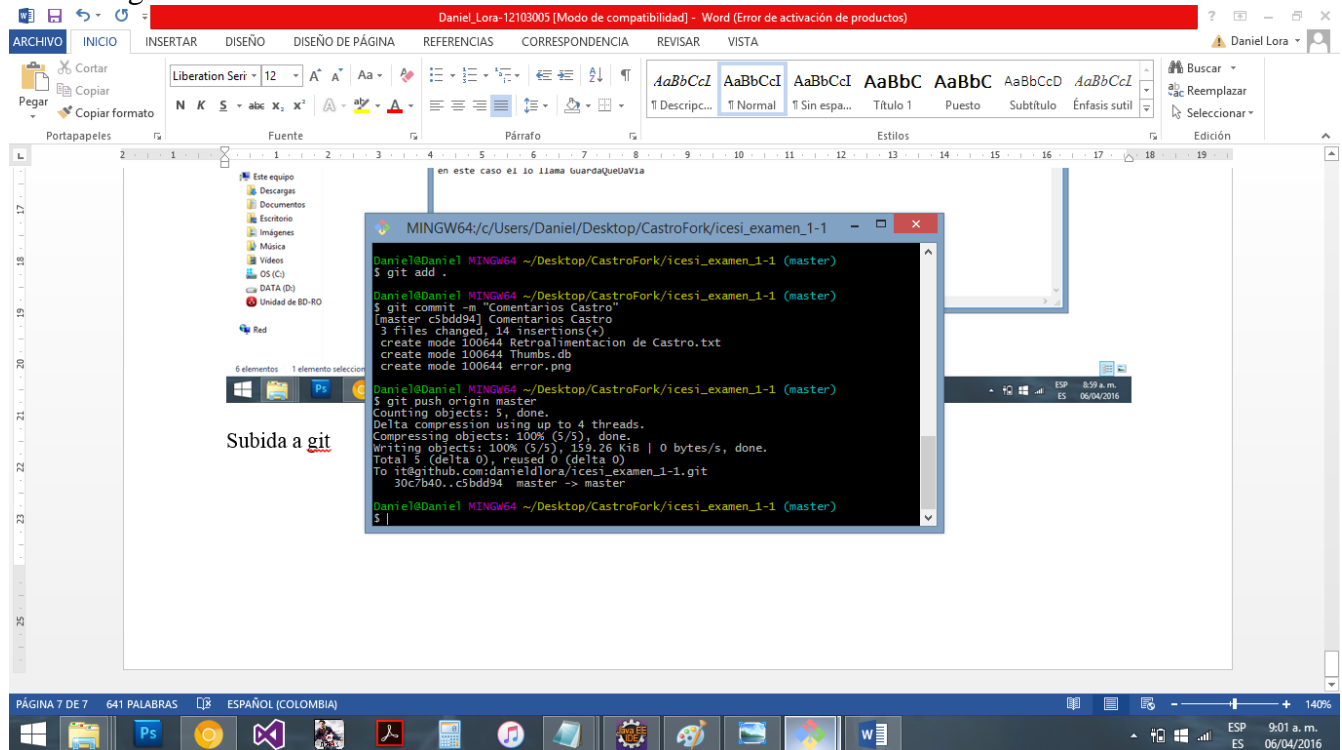
Git clone del repositorio



Retroalimentacion



Subida a git



Git pull request

