

Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica
IE-0117 Programación Bajo Plataformas Abiertas
III-ciclo 2022

Laboratorio

GIT

Aroon Sanabria Torres B97205
Grupo 901

Profesor: Julián Gairaud

10 de febrero de 2023

Índice

1. Descripción	1
1.1. Creación de un fork del repositorio en el repositorio personal.	1
1.2. Creación de un nuevo branch.	1
1.3. Cambio del objeto gato1 desde main.	4
1.4. Inclusión de feature en main.	4
1.5. Captura de los conflictos entre ambas ramas.	5
1.6. Solución del conflicto.	5
1.7. Inclusión de los cambios al repositorio.	5
1.8. Manual de git log.	6
1.9. Archivo gatos.py corriendo en la computadora.	6
1.10. Commit en el repositorio local.	7
1.11. Push de la rama main local a la rama main del repositorio remoto.	7

Índice de figuras

1.	Creación de un fork del repositorio marpermon hacia arsanabria.	1
2.	Clonación del repositorio con el comando <i>git clone</i> en el ordenador personal. . .	1
3.	Creación y cambio a una rama con el comando <i>git branch</i> y <i>git checkout</i>	1
4.	Edición del nombre gato1 con el comando <i>nano</i> desde feature.	2
5.	Creación de un nuevo objeto gato3 en el feature con los comandos, <i>git add</i> y <i>git commit</i>	2
6.	Creación de un nuevo objeto gato3 en el feature con el comando <i>nano</i>	2
7.	Envío del branch feature hacia el branch remoto con el comando <i>git push</i>	3
8.	Validación del envío del branch feature al branch remoto.	3
9.	Validación del cambio del branch feature al branch remoto.	3
10.	Cambio a la rama main con el comando <i>git checkout</i> y se añade al stage con el comando <i>add</i> y su debido commit.	4
11.	Edición del objeto gato1 desde main con el comando <i>nano</i>	4
12.	Conflicto de contenido entre ambas ramas.	5
13.	Solución del conflicto con los comandos <i>git add</i> y <i>git rebase continue</i>	5
14.	Inclusión de los cambios al repositorio con el comando <i>git push</i>	5
15.	Uso del comando <i>git log oneline graph</i>	6
16.	Manual de graph.	6
17.	Significado de oneline.	6
18.	Función de oneline	6
19.	Ejecución del archivo gatos.py.	6
20.	Creación de un commit en el repositorio main con el pdf de los puntos 5,7,8 y 9.	7
21.	Push de la rama main local a la rama main del repositorio remoto.	7

1. Descripción

En el repositorio público, cuyo link es: <https://github.com/marpermon/LaboratorioGit> se encuentra una carpeta llamada `gatosGit`, con dos archivos, uno llamado `funcionesTarea.py`, el cual NO se debe modificar, y otro llamado `gatos.py`. Las líneas de código se crea un objeto llamado `gato1` y el segundo como `gato2`.

1.1. Creación de un fork del repositorio en el repositorio personal.

Se dirige hacia el "fork" de repositorio `marpermon` y se crea uno de ellos de manera publica con el nombre de `LaboratorioGit` como se muestra en la figura 1.

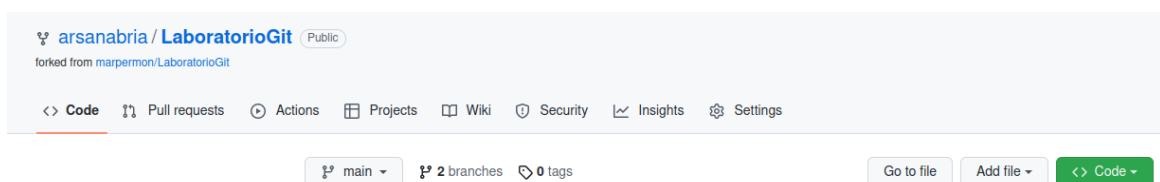


Figura 1: Creación de un fork del repositorio `marpermon` hacia `arsanabria`.

```
arsanabria@ie0117:~$ git clone git@github.com:arsanabria/LaboratorioGit.git
Cloning into 'LaboratorioGit'...
remote: Enumerating objects: 19, done.
remote: Counting objects: 100% (19/19), done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 19 (delta 2), reused 19 (delta 2), pack-reused 0
Receiving objects: 100% (19/19), done.
Resolving deltas: 100% (2/2), done.
arsanabria@ie0117:~$
```

Figura 2: Clonación del repositorio con el comando `git clone` en el ordenador personal.

1.2. Creación de un nuevo branch.

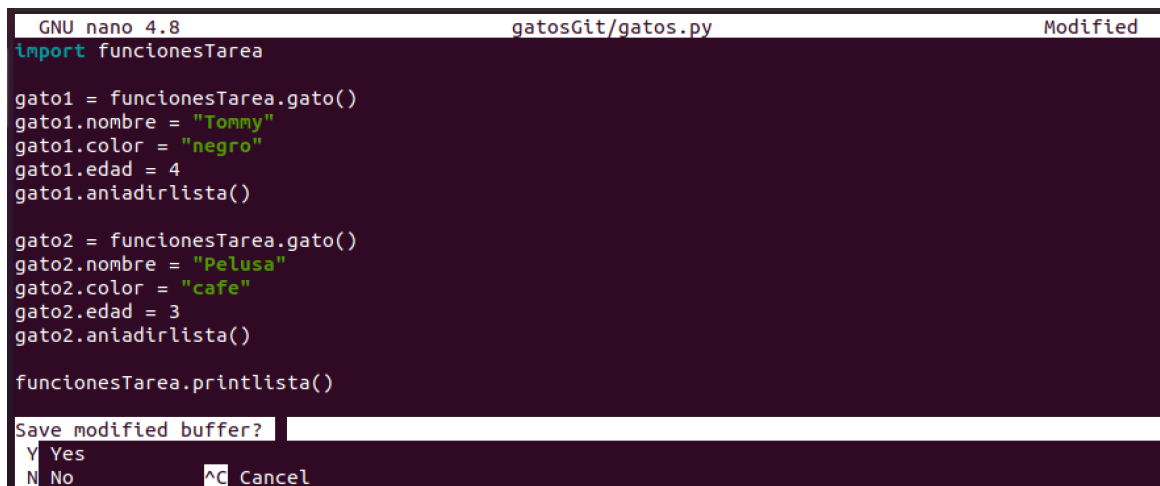
```
arsanabria@ie0117:~/LaboratorioGit$ git branch feature
arsanabria@ie0117:~/LaboratorioGit$ git checkout feature
Switched to branch 'feature'
arsanabria@ie0117:~/LaboratorioGit$ sudo nano gatosGit/gatos.py
[sudo] password for arsanabria:
arsanabria@ie0117:~/LaboratorioGit$ git add gatosGit/gatos.py
arsanabria@ie0117:~/LaboratorioGit$ git commit -m "Cambio del nombre del gato1 desde feature"
[feature 76500e9] Cambio del nombre del gato1 desde feature
1 file changed, 1 insertion(+), 1 deletion(-)
arsanabria@ie0117:~/LaboratorioGit$ git log
commit 76500e9d18022e2a37eb713d1d7b0066cf2f5e0a (HEAD -> feature)
Author: arsanabria <ar.sanabriatorres@gmail.com>
Date: Fri Feb 10 04:50:12 2023 -0600

    Cambio del nombre del gato1 desde feature

commit 1640128b5cd29d26384cbfa07ba2ceca7e336d21 (origin/main, origin/HEAD, main)
Author: marpermon <maria.perezmonge@ucr.ac.cr>
Date: Sat Jan 14 14:17:16 2023 -0600
```

Figura 3: Creación y cambio a una rama con el comando `git branch` y `git checkout`.

Además se utiliza el comando *git commit -m* para crear un commit con los cambios en estado stage y *git log* para ver el historial de commimts.



```
GNU nano 4.8                                gatosGit/gatos.py                Modified
import funcionesTarea

gato1 = funcionesTarea.gato()
gato1.nombre = "Tommy"
gato1.color = "negro"
gato1.edad = 4
gato1.aniadirlista()

gato2 = funcionesTarea.gato()
gato2.nombre = "Pelusa"
gato2.color = "cafe"
gato2.edad = 3
gato2.aniadirlista()

funcionesTarea.printlista()

Save modified buffer? [Y] Yes
[ N ] No                  [^C] Cancel
```

Figura 4: Edición del nombre gato1 con el comando *nano* desde feature.



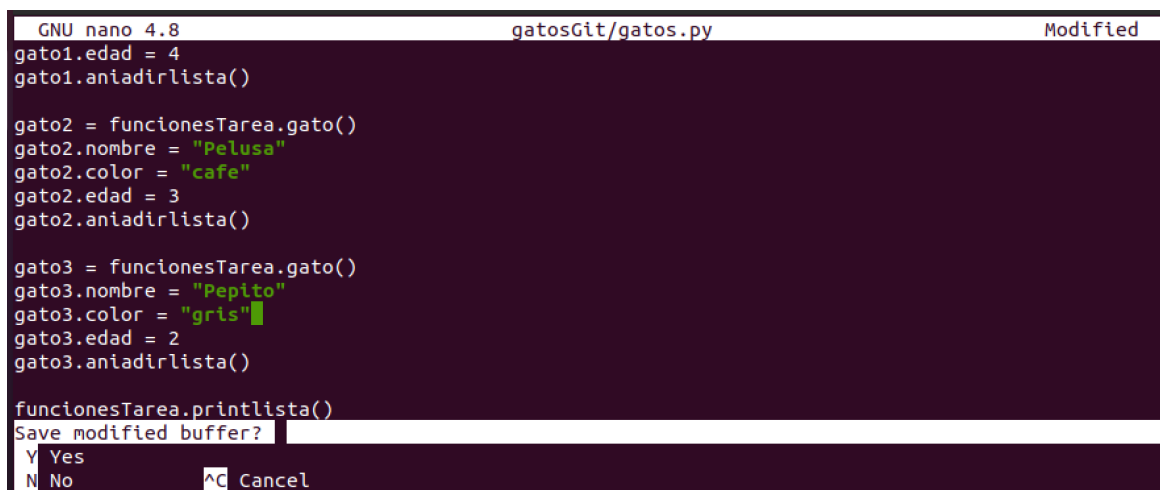
```
arsanabria@ie0117:~/LaboratorioGit$ nano gatosGit/gatos.py
arsanabria@ie0117:~/LaboratorioGit$ git add gatosGit/gatos.py
arsanabria@ie0117:~/LaboratorioGit$ git commit -m "Creacion del gato3 desde el feature"
[feature 5cb9c52] Creacion del gato3 desde el feature
 1 file changed, 6 insertions(+)
arsanabria@ie0117:~/LaboratorioGit$ git log
commit 5cb9c521f8814c6a75ee89c5c13a6484e262b0db (HEAD -> feature)
Author: arsanabria <ar.sanabriatorres@gmail.com>
Date:   Fri Feb 10 04:57:08 2023 -0600

    Creacion del gato3 desde el feature

commit 76500e9d18022e2a37eb713d1d7b0066cf2f5e0a
Author: arsanabria <ar.sanabriatorres@gmail.com>
Date:   Fri Feb 10 04:50:12 2023 -0600

    Cambio del nombre del gato1 desde feature
```

Figura 5: Creación de un nuevo objeto gato3 en el feature con los comandos, *git add* y *git commit*.



```
GNU nano 4.8                                gatosGit/gatos.py                Modified
gato1.edad = 4
gato1.aniadirlista()

gato2 = funcionesTarea.gato()
gato2.nombre = "Pelusa"
gato2.color = "cafe"
gato2.edad = 3
gato2.aniadirlista()

gato3 = funcionesTarea.gato()
gato3.nombre = "Pepito"
gato3.color = "gris"
gato3.edad = 2
gato3.aniadirlista()

funcionesTarea.printlista()

Save modified buffer? [Y] Yes
[ N ] No                  [^C] Cancel
```

Figura 6: Creación de un nuevo objeto gato3 en el feature con el comando *nano* .

```

arsanabria@ie0117:~/LaboratorioGit$ git push origin feature:feature
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 4 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (8/8), 840 bytes | 840.00 KiB/s, done.
Total 8 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
remote:
remote: Create a pull request for 'feature' on GitHub by visiting:
remote:   https://github.com/arsanabria/LaboratorioGit/pull/new/feature
remote:
To github.com:arsanabria/LaboratorioGit.git
 * [new branch]      feature -> feature
arsanabria@ie0117:~/LaboratorioGit$

```

Figura 7: Envío del branch feature hacia el branch remoto con el comando *git push*.

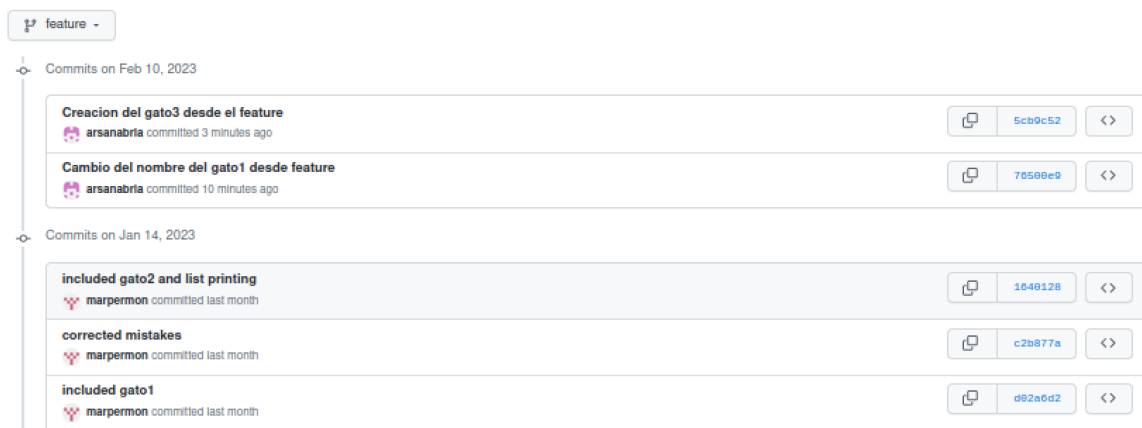


Figura 8: Validación del envío del branch feature al branch remoto.

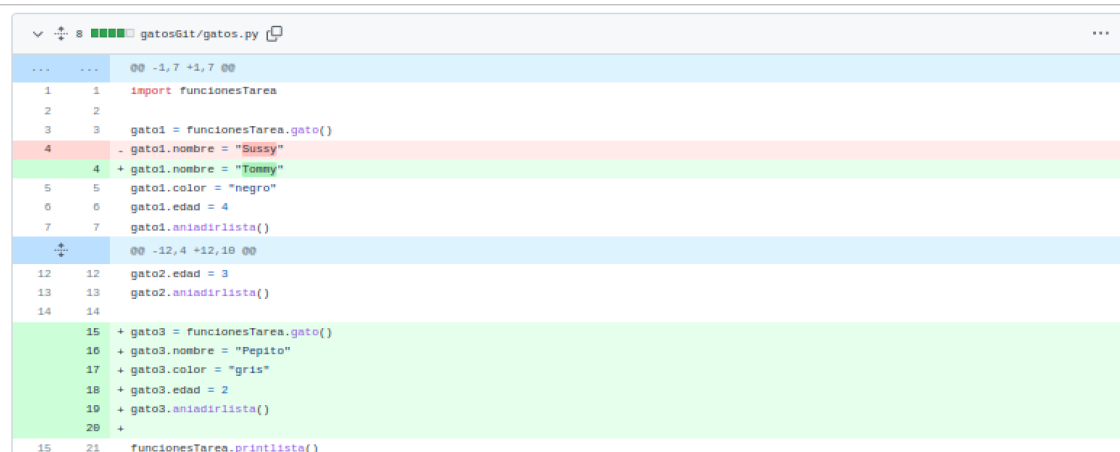
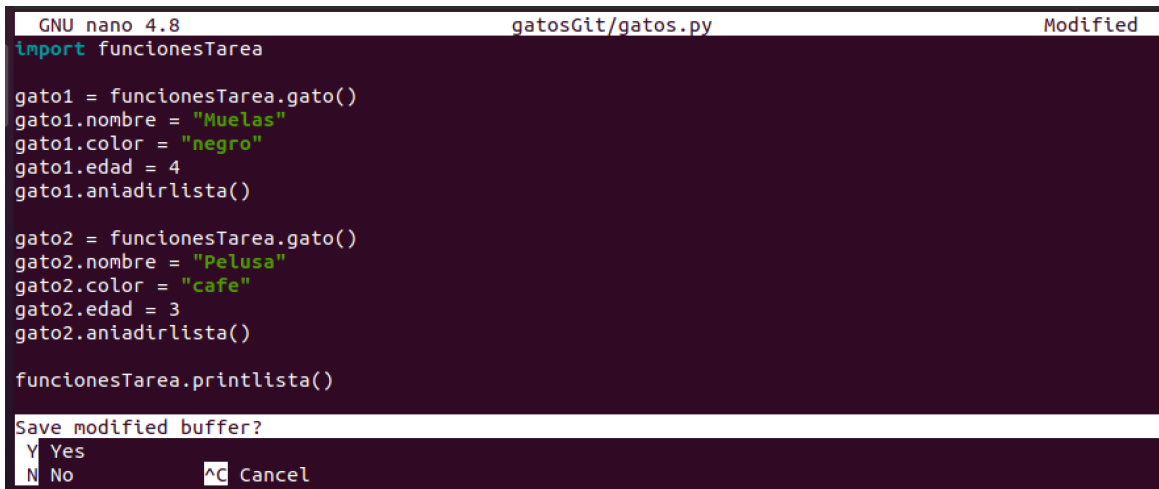


Figura 9: Validación del cambio del branch feature al branch remoto.

1.3. Cambio del objeto gato1 desde main.

```
arsanabria@ie0117:~/LaboratorioGit$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
arsanabria@ie0117:~/LaboratorioGit$ nano gatosGit/gatos.py
arsanabria@ie0117:~/LaboratorioGit$ git add gatosGit/gatos.py
arsanabria@ie0117:~/LaboratorioGit$ git commit -m "Cambio del nombre del gato1 desde main"
[main cec55f0] Cambio del nombre del gato1 desde main
1 file changed, 1 insertion(+), 1 deletion(-)
arsanabria@ie0117:~/LaboratorioGit$
```

Figura 10: Cambio a la rama main con el comando *git checkout* y se añade al stage con el comando *add* y su debido commit.



```
GNU nano 4.8                                gatosGit/gatos.py                                Modified
import funcionesTarea

gato1 = funcionesTarea.gato()
gato1.nombre = "Muelas"
gato1.color = "negro"
gato1.edad = 4
gato1.aniadirlista()

gato2 = funcionesTarea.gato()
gato2.nombre = "Pelusa"
gato2.color = "cafe"
gato2.edad = 3
gato2.aniadirlista()

funcionesTarea.printlista()

Save modified buffer?
Y Yes
N No      ^C Cancel
```

Figura 11: Edición del objeto gato1 desde main con el comando *nano*

1.4. Inclusión de feature en main.

Se realiza rebase al feature desde main con el comando *git fetch* y *git rebase* como se muestra en la figura 12.

1.5. Captura de los conflictos entre ambas ramas.

```
arsanabria@ie0117:~/LaboratorioGit$ git fetch origin feature
From github.com:arsanabria/LaboratorioGit
* branch       feature    -> FETCH_HEAD
arsanabria@ie0117:~/LaboratorioGit$ git rebase origin/feature
First, rewinding head to replay your work on top of it...
Applying: Cambio del nombre del gato1 desde main
Using index info to reconstruct a base tree...
M       gatosGit/gatos.py
Falling back to patching base and 3-way merge...
Auto-merging gatosGit/gatos.py
CONFLICT (content): Merge conflict in gatosGit/gatos.py
error: Failed to merge in the changes.
Patch failed at 0001 Cambio del nombre del gato1 desde main
hint: Use 'git am --show-current-patch' to see the failed patch
Resolve all conflicts manually, mark them as resolved with
"git add/rm <conflicted_files>", then run "git rebase --continue".
You can instead skip this commit: run "git rebase --skip".
To abort and get back to the state before "git rebase", run "git rebase --abort".
arsanabria@ie0117:~/LaboratorioGit$
```

Figura 12: Conflicto de contenido entre ambas ramas.

1.6. Solución del conflicto.

```
arsanabria@ie0117:~/LaboratorioGit$ git add gatosGit/gatos.py
arsanabria@ie0117:~/LaboratorioGit$ git rebase --continue
Applying: Cambio del nombre del gato1 desde main
arsanabria@ie0117:~/LaboratorioGit$
```

Figura 13: Solución del conflicto con los comandos *git add* y *git rebase continue*.

1.7. Inclusión de los cambios al repositorio.

```
arsanabria@ie0117:~/LaboratorioGit$ git status
On branch main
Your branch is ahead of 'origin/main' by 3 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
arsanabria@ie0117:~/LaboratorioGit$ git push
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 496 bytes | 496.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:arsanabria/LaboratorioGit.git
   1640128..469e117  main -> main
arsanabria@ie0117:~/LaboratorioGit$
```

Figura 14: Inclusión de los cambios al repositorio con el comando *git push*.


```

arsanabria@ie0117:~/LaboratorioGit$ git log --oneline --graph
* 469e117 (HEAD -> main, origin/main, origin/HEAD) Cambio del nombre del gato1 desde main
* 5cb9c52 (origin/feature, feature) Creacion del gato3 desde el feature
* 76500e9 Cambio del nombre del gato1 desde feature
* 1640128 included gato2 and list printing
* c2b877a corrected mistakes
* d02a6d2 included gato1
* 7ab0adc included functions file
* a13fd08 first commit
arsanabria@ie0117:~/LaboratorioGit$

```

Figura 15: Uso del comando *git log oneline graph*.

1.8. Manual de git log.

```

--graph
    Draw a text-based graphical representation of the commit history on the left hand
    side of the output. This may cause extra lines to be printed in between commits, in
    order for the graph history to be drawn properly. Cannot be combined with
    --no-walk.

    This enables parent rewriting, see History Simplification above.

    This implies the --topo-order option by default, but the --date-order option may
    also be specified.

```

Figura 16: Manual de graph.

```

--oneline
    This is a shorthand for "--pretty=oneline --abbrev-commit" used together.

```

Figura 17: Significado de oneline.

```

• oneline

    <hash> <title line>

    This is designed to be as compact as possible.

```

Figura 18: Función de oneline

1.9. Archivo gatos.py corriendo en la computadora.

```

arsanabria@ie0117:~/LaboratorioGit/gatosGit$ vim gatos.py
arsanabria@ie0117:~/LaboratorioGit/gatosGit$ python3 gatos.py
Muelas, 4 annos, color: negro
Pelusa, 3 annos, color: cafe
Pepito, 2 annos, color: gris
arsanabria@ie0117:~/LaboratorioGit/gatosGit$

```

Figura 19: Ejecución del archivo gatos.py.

1.10. Commit en el repositorio local.

```
arsanabria@ie0117:~/LaboratorioGit$ ls
gatosGit LaboratorioGIT_Paso10_B97205.pdf README.md
arsanabria@ie0117:~/LaboratorioGit$ git add LaboratorioGIT_Paso10_B97205.pdf
arsanabria@ie0117:~/LaboratorioGit$ git commit -m "Pdf de los puntos 5,7,8 y 9"
[main 495c8ad] Pdf de los puntos 5,7,8 y 9
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 LaboratorioGIT_Paso10_B97205.pdf
```

Figura 20: Creación de un commit en el repositorio main con el pdf de los puntos 5,7,8 y 9.

1.11. Push de la rama main local a la rama main del repositorio remoto.

```
arsanabria@ie0117:~/LaboratorioGit$ git push origin feature
Everything up-to-date
arsanabria@ie0117:~/LaboratorioGit$ git log --oneline --graph
* 495c8ad (HEAD -> main) Pdf de los puntos 5,7,8 y 9
* 469e117 (origin/main, origin/HEAD) Cambio del nombre del gato1 desde main
* 5cb9c52 (origin/feature, feature) Creacion del gato3 desde el feature
* 76500e9 Cambio del nombre del gato1 desde feature
* 1640128 included gato2 and list printing
* c2b877a corrected mistakes
* d02a6d2 included gato1
* 7ab0adc included functions file
* a13fd08 first commit
arsanabria@ie0117:~/LaboratorioGit$
```

Figura 21: Push de la rama main local a la rama main del repositorio remoto.