En cas de conflit :

Le conflit apparaît lors du push via ce type de message :

```
Serge@DESKTOP-QEGHFIP MINGW64 ~/Documents/Maths/L3/3M101-Quadtree (master)
$ git push
To https://github.com/sergedurand/3M101-Quadtree.git
 ! [rejected]        master -> master (fetch first)
error: failed to push some refs to 'https://github.com/sergedurand/3M101-Quadtree.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

On suit le message et on fait “git pull”, qui donne ce message :

```
Serge@DESKTOP-QEGHFIP MINGW64 ~/Documents/Maths/L3/3M101-Quadtree (master)
$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/sergedurand/3M101-Quadtree
   497849f..3883d84  master     -> origin/master
Auto-merging commandes git divers.txt
CONFLICT (content): Merge conflict in commandes git divers.txt
Automatic merge failed; fix conflicts and then commit the result.
```

Encadré en rouge on voit le fichier où il y a conflit.

Il faut ensuite d’ouvrir le fichier en question avec un éditeur de texte classique, on y voit quelque chose de ce genre :

<<<<<<< HEAD
autre test

=======
retest
>>>>>>> 3883d849899f649bd8339448acf49774671451e4

Ce qui est situé juste après HEAD est la partie que vous vouliez ajouter au document, ce qui est situé au dessous du “=======” est la partie qui a été ajouté en même temps.
Le 3883d849899f649bd8339448acf49774671451e4 c’est l’ID du commit qui a fait cet ajout.

Il suffit ensuite de résoudre le conflit à la main direct dans l'éditeur : si les deux ajouts concernent des parties différentes et qu'on veut garder les deux, il suffit de supprimer tout ce qui a été ajouté par GIT et de conserver juste les modifs, on ne garde que :

autre test

retest

(En modifiant l'ordre si nécessaire.) Si les modifs portaient sur la même portion / phrase faut faire un choix, garder la meilleure modif des deux, réécrire la phrase pour mixer les deux modifs… ça ne devrait pas arriver souvent en tout cas.

Une fois que le fichier a été modifié pour contenir sa version finale on indique à git qu'on a géré le conflit en refaisant un commit :

"git add * "
"git commit -m "conflit résolu" (ou n'importe quel autre message…)
"git push"