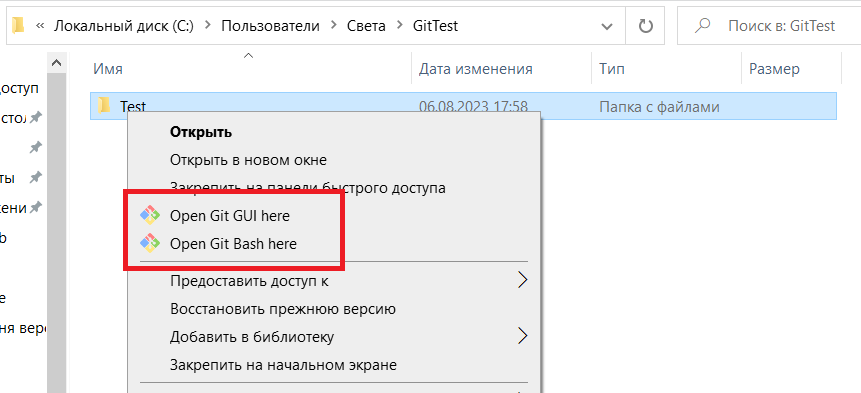
**Оновлення даних, синхронізація віддаленого і локального репозиторія**

Зміни ми бачимо тільки у GitHub, на локальному комп'ютері в Git ми цього не побачимо, тому на треба дещо зробити:

На локальному комп'ютері ми можемо працювати з КМ, а також через викликане контекстне меню відкрити через «Open Git Bash here»





**1. Після внесених змін на GitHub наш Git цих змін не бачить, ми можемо це перевірити командою «cat file1.txt» (це назва нашого файла) ця команда дійсна у «Open Git Bash here» та Linux :**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ **cat file1.txt**

123

456

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$

Тут нам зараз показує дані до внесення змін

**2. Для того щоб подивитися чи є зміни треба задати команду “git fetch”:**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ **git fetch**

remote: Enumerating objects: 5, done.

remote: Counting objects: 100% (5/5), done.

remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (3/3), 648 bytes | 36.00 KiB/s, done.

From https://github.com/Svetochek30/Test

e888883..7fc2837 main -> origin/main

**3. Тепер якщо ми задамо команду “git status”, то у відповіді нам покаже на скільки коммітів ми відстаємо від серверної версії**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ **git status**

On branch main

Your branch is behind 'origin/main' by 1 commit, and can be fast-forwarded.

(use "git pull" to update your local branch)

nothing to commit, working tree clean

**4. Для того щоб оновити дані у Git після зміни даних у Git Hub треба виконати команду "git pull"**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ **git pull**

Updating e888883..453ba65

Fast-forward

File1.txt | 4 +++-

1 file changed, 3 insertions(+), 1 deletion(-)

**5. Якщо на треба подивитися які зміни вносила конкретна людина, треба ввести команду «git log --author ім'я людини»**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ **git log --author Svetochek30**

commit 453ba653574e341c60b3704e82d741eaa4155af5 (HEAD -> main, origin/main)

Author: Svetochek30 <141569185+Svetochek30@users.noreply.github.com>

Date: Wed Aug 9 12:36:41 2023 +0300

Update File1.txt

add

commit 7fc2837c76f999400b8b41431e5a75489e829672

Author: Svetochek30 <141569185+Svetochek30@users.noreply.github.com>

Date: Mon Aug 7 20:23:07 2023 +0300

Update File1.txt

Add a new line

**6.Якщо нам треба подивитися інформацію по конкретному комміті треба ввести команду**

**git show (інтетифікаційний номер комміта – скопіювати)**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ **git show 453ba653574e341c60b3704e82d741eaa4155af5**

commit 453ba653574e341c60b3704e82d741eaa4155af5 (HEAD -> main, origin/main)

Author: Svetochek30 <141569185+Svetochek30@users.noreply.github.com>

Date: Wed Aug 9 12:36:41 2023 +0300

Update File1.txt

add

diff --git a/File1.txt b/File1.txt

index 2ccfe33..55c15f3 100644

--- a/File1.txt

+++ b/File1.txt

@@ -2,4 +2,4 @@

456

-789.

+7897.

**7. Якщо ввести команду «git show» без індетифікаціного номеру, то нам видасть інформацію по останньому зміненому комміті**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ **git show**

commit 453ba653574e341c60b3704e82d741eaa4155af5 (HEAD -> main, origin/main)

Author: Svetochek30 <141569185+Svetochek30@users.noreply.github.com>

Date: Wed Aug 9 12:36:41 2023 +0300

Update File1.txt

add

diff --git a/File1.txt b/File1.txt

index 2ccfe33..55c15f3 100644

--- a/File1.txt

+++ b/File1.txt

@@ -2,4 +2,4 @@

456

-789.

+7897.

**8. Якщо нам треба дізнатися інформаці про автора конкретного файлу**

**git blame (назва файлу)**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ **git blame File1.txt**

^e888883 (unknown 2023-08-06 18:18:56 +0300 1) 123

^e888883 (unknown 2023-08-06 18:18:56 +0300 2)

7fc2837c (Svetochek30 2023-08-07 20:23:07 +0300 3) 456

7fc2837c (Svetochek30 2023-08-07 20:23:07 +0300 4)

453ba653 (Svetochek30 2023-08-09 12:36:41 +0300 5) 7897.

**9. Якщо нам треба дізнатися хто саме вносив зміни у конкретних рядках**

**git blame (назва файлу) | grep (перелік рядків)**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git blame File1.txt | grep 123

^e888883 (unknown 2023-08-06 18:18:56 +0300 1) 123

**10. Якщо ми задамо команду перевірки статусу, то на покаже дані оновлення до поточної дати**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ **git status**

On branch main

Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

**11. Щоб перевірити оновлення задаємо команду «cat file1.txt»**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ cat file1.txt

123

456

7897.

Оновлення є, це означає що віддалений і локальний репозиторій синхронізовані

**Зміни даних у локальному файлі через**

«Open Git Bash here»

**1. Для того щоб додати рядок у локальний файл через** «Open Git Bash here» **треба ввести команду echo "Add Line" >> назва файлу**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ **echo "Add Line" >> File1.txt**

**2. Щоб перевірити чи додався рядок, використаємо команду**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ **cat File1.txt**

123

456

7897.

Add Line

Ми бачимо, що рядок додався

**3. Якщо ввести команду «git diff», то нам покаже які саме зміни відбулися**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ **git diff**

warning: in the working copy of 'File1.txt', LF will be replaced by CRLF the next time Git touches it

diff --git a/File1.txt b/File1.txt

index 55c15f3..e7c5cb5 100644

--- a/File1.txt

+++ b/File1.txt

@@ -3,3 +3,4 @@

456

7897.

+Add Line

**Робота з Git commit**

**1. Якщо ми запустимо команду «ls», то нам покаже назву нашого файлу проекту**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ **ls**

File1.txt

**2. Якщо ми запустимо команду «cat File1.txt», то нам покаже що в цьому файлі**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ **cat File1.txt**

123

456

7897.

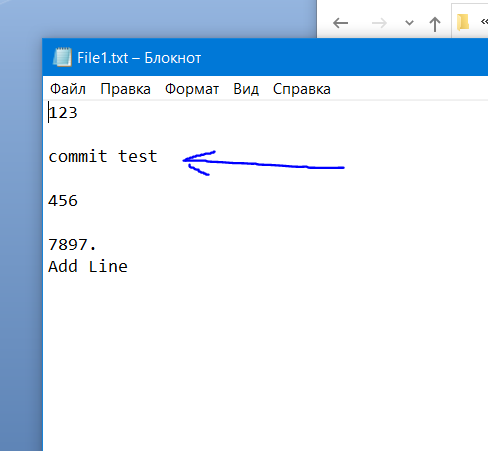
Add Line

**3.Щоб створити новий файл треба задати команду** **echo "new line" >> new.file**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ **echo "new line" >> new.file**

І додатко у файлі вручну добавимо рядок



**4.Тепер щоб перевірити зміни треба задати команду**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: File1.txt

Untracked files:

(use "git add <file>..." to include in what will be committed)

new.file

no changes added to commit (use "git add" and/or "git commit -a")

**5. Зараз ми просто додамо зміни (комміт) задавши команду**

«git commit -a -m "new commit»

Де:

«-a» - це позначка яка показує, до ми додаємо всі зміни (якщо треба додати зміни у конкретному файлі, то пишемо його назву;

«-m» - це меседж, тобто коментар, що саме міститься в цьому комміті (змінах).

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git commit -a -m "new commit"

warning: in the working copy of 'File1.txt', LF will be replaced by CRLF the nex

t time Git touches it

[main 936374b] new commit

1 file changed, 2 insertions(+)

Тепер нам показує,що зміни додано, але файл ні, це тому, що ми його не додали у репозиторій, а зміни відбулися вже у доданому файлі

**6. Якщо нам треба відмінити зміни у коміті ми вокористовуємо команду**

git reset HEAD~1

Де «HEAD~1» показує на скільки кроків треба відмінити зміни (це видалення буде безповоротним, повернути його не можливо)

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git reset HEAD~1

Unstaged changes after reset:

M File1.txt

**7. Тепер якщо ми перевіримо стату, то нам покаже стан наших файлів до того, як ми виконали дадати зміни**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: File1.txt

Untracked files:

(use "git add <file>..." to include in what will be committed)

new.file

no changes added to commit (use "git add" and/or "git commit -a")

**8. Повернемося до стану де ми додали комміт, щоб подивитися, що ще можна зробити**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git commit -a -m "new commit"

warning: in the working copy of 'File1.txt', LF will be replaced by CRLF the nex

t time Git touches it

[main ae6252d] new commit

1 file changed, 2 insertions(+)

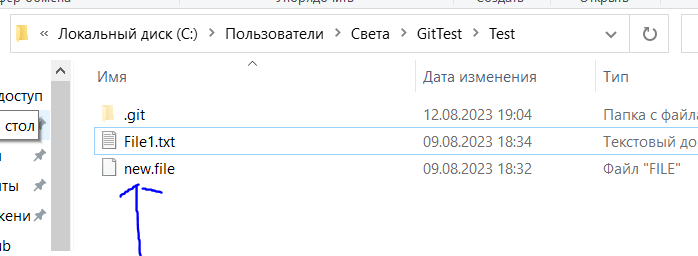
**9. Додамо додатковий файл у репозиторій**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git add new.file

warning: in the working copy of 'new.file', LF will be replaced by CRLF the next

time Git touches it



**10. Якщо ми перевіремо тепер статус, то додатковий (новий) теж додасться**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git status

On branch main

Your branch is ahead of 'origin/main' by 1 commit.

(use "git push" to publish your local commits)

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

new file: new.file

**11. Тепер його треба закоммітити**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git commit -a -m "test"

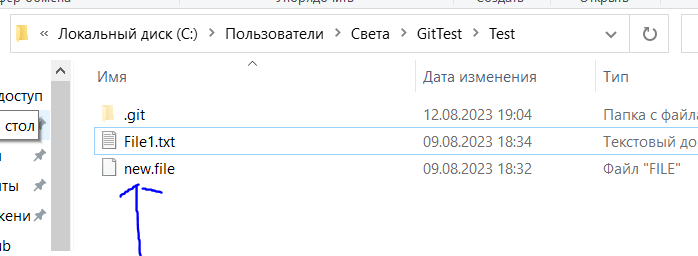
[main 715a296] test

1 file changed, 1 insertion(+)

create mode 100644 new.file

**Конфлікти**

**Конфлікт 1.** Тепер щоб нам бачити цей файл на віддаленому репозиторії треба виконати команду «git push», але бувають ситуації коли одночасно створюють ще файл у віддаленому репозиторіії і так як ми його ще не додали до локального, а наш змінений не додали до віддаленого виникне конфлікт, як це вирішити:



**1. Якщо зараз ми використаємо команду «git push» то нам видасть помилку**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git push

To https://github.com/Svetochek30/Test.git

! [rejected] main -> main (fetch first)

error: failed to push some refs to 'https://github.com/Svetochek30/Test.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.

**2. Для того щоб це виправити, ми спочатку додамо файл з віддаленогорепозиторія на локальний**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git pull

remote: Enumerating objects: 4, done.

remote: Counting objects: 100% (4/4), done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (3/3), 653 bytes | 11.00 KiB/s, done.

From https://github.com/Svetochek30/Test

182da48..10f6777 main -> origin/main

Merge made by the 'ort' strategy.

conflict | 1 +

1 file changed, 1 insertion(+)

create mode 100644 conflict

**3. А тепер з локального на віддалений**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git push

Enumerating objects: 12, done.

Counting objects: 100% (11/11), done.

Delta compression using up to 4 threads

Compressing objects: 100% (5/5), done.

Writing objects: 100% (8/8), 855 bytes | 855.00 KiB/s, done.

Total 8 (delta 0), reused 0 (delta 0), pack-reused 0

To https://github.com/Svetochek30/Test.git

10f6777..2684f90 main -> main

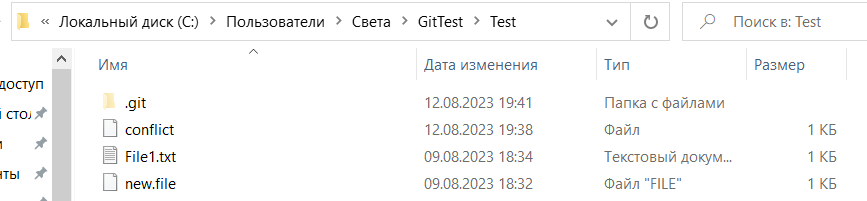
**4. Перевіряємо командою git pull , що все оновился**

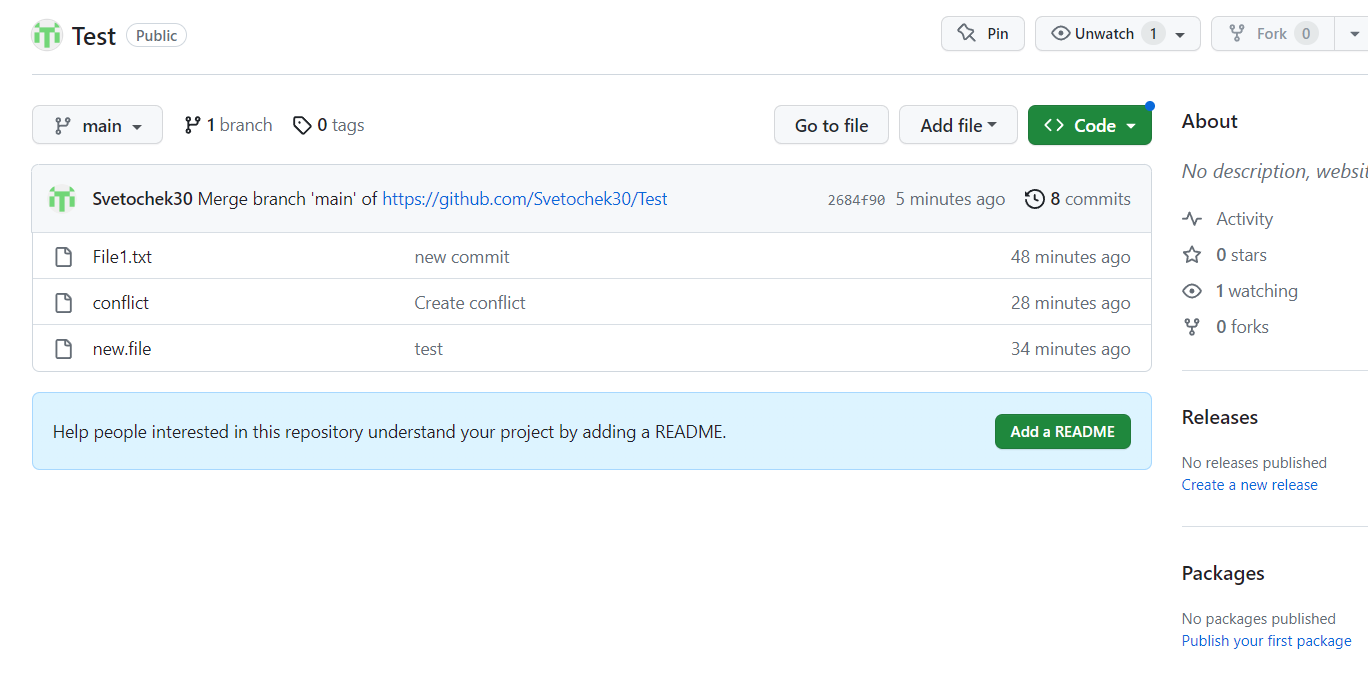
Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git pull

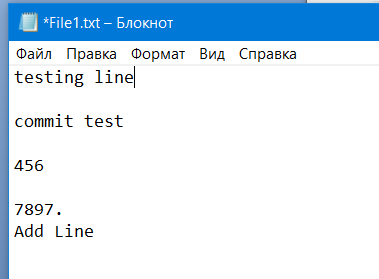
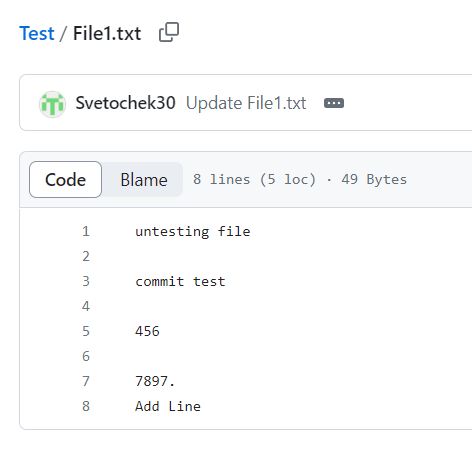
Already up to date.

Тепер все добре, иформація на локальному і віддаленому репозиторії вирівнялися





**Конфлікт 2.** Ще може виникнути ситуація коли були здійснені виправлення в одному і томуж файлі, в одному і томуж рядку як у локальному, так і віддаленому репозиторії

**1. Спробуємо все закоммітити на локальному комп'ютері**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git commit -am "test"

warning: in the working copy of 'File1.txt', LF will be replaced by CRLF the nex

t time Git touches it

[main 7e71f4c] test

1 file changed, 1 insertion(+), 1 deletion(-)

Все вийшло

**2. Але при спробі залити з локального у віддалений видасть помилку**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git push

To https://github.com/Svetochek30/Test.git

! [rejected] main -> main (fetch first)

error: failed to push some refs to 'https://github.com/Svetochek30/Test.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.

**3. Та при спробі залити навпаки з віддаленого у локальний, теж виникне конфлікт**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git pull

remote: Enumerating objects: 5, done.

remote: Counting objects: 100% (5/5), done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (3/3), 743 bytes | 46.00 KiB/s, done.

From https://github.com/Svetochek30/Test

2684f90..cf73914 main -> origin/main

Auto-merging File1.txt

**CONFLICT (content): Merge conflict in File1.txt**

Automatic merge failed; fix conflicts and then commit the result.

**4. Що з цим робити? Якщо в нас немає доступу, чи ми не знаємо як і що змінювалося у файлі, тоді ми звертиємося до розробників і вони вже вирішують це. І якщо нам далі треба працювати для цього є команда git merge –abort, після неї зміни яка нам давала конфлікт не покажеться**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main|MERGING)

$ git merge --abort

Тепер перевірити статус

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git status

On branch main

Your branch and 'origin/main' have diverged,

and have 1 and 1 different commits each, respectively.

(use "git pull" to merge the remote branch into yours)

nothing to commit, working tree clean

**5. Але ми можемо це вирішити, вводимо команду git pull, потім git status**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git pull

Auto-merging File1.txt

CONFLICT (content): Merge conflict in File1.txt

Automatic merge failed; fix conflicts and then commit the result.

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main|MERGING)

$ git status

On branch main

Your branch and 'origin/main' have diverged,

and have 1 and 1 different commits each, respectively.

(use "git pull" to merge the remote branch into yours)

You have unmerged paths.

(fix conflicts and run "git commit")

(use "git merge --abort" to abort the merge)

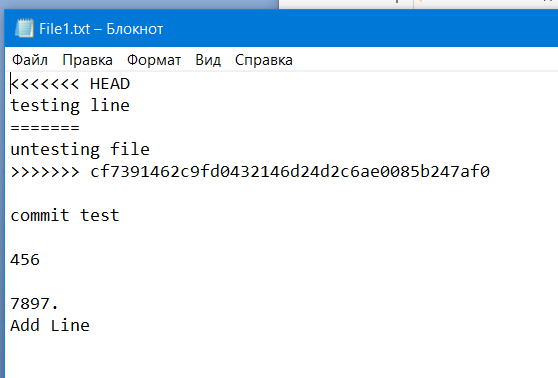
Unmerged paths:

(use "git add <file>..." to mark resolution)

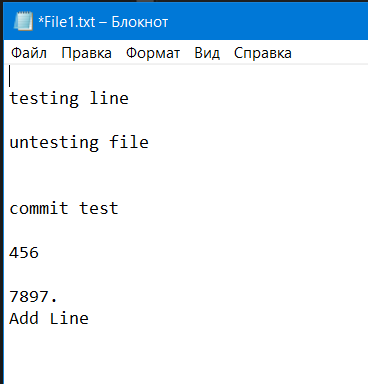
both modified: File1.txt

no changes added to commit (use "git add" and/or "git commit -a")

. . . тепер на локальному комп'ютері заходимоу у файл у якого є конфліт, і бачимо що система помітила нам місто де є конфлікт. . .



. . . тепер ми редактуємо як все повинно залишитися, наприклад ми хочемо залишити обидва рядки, тоді просто видаляємо зайву інформацію



. . . тепер треба завержити ці зміни командами

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main|MERGING)

$ git add .

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main|MERGING)

$ git commit -am "test"

[main 17b0874] test

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git push

Enumerating objects: 10, done.

Counting objects: 100% (10/10), done.

Delta compression using up to 4 threads

Compressing objects: 100% (5/5), done.

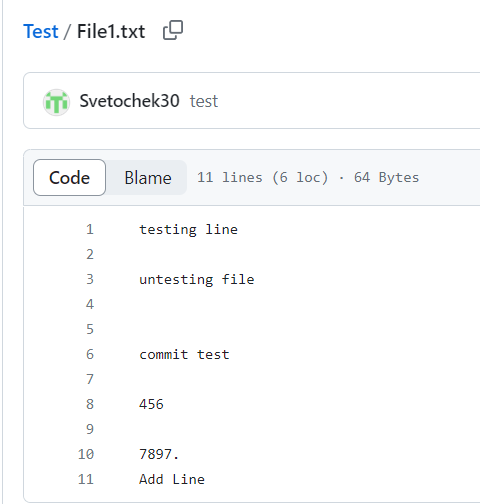
Writing objects: 100% (6/6), 700 bytes | 700.00 KiB/s, done.

Total 6 (delta 0), reused 0 (delta 0), pack-reused 0

To https://github.com/Svetochek30/Test.git

cf73914..17b0874 main -> main

. . .все добре все оновилося



. . . тепер перевіримо на локальному репозиторії

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ cat File1.txt

testing line

untesting file

commit test

456

7897.

Add Line

**Відкат файлів**

Як це зробити:

**1. Відкат в одному файлі**

Додамо новий рядок у локальному репозиторії. . .

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ echo "3line" >> File1.txt

. . .перевіремо

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ cat File1.txt

testing line

untesting file

commit test

456

7897.

Add Line

3line

. . . бачимо що дабавилося, тепер ми можемо виконати команду і обов'язково написати файл який ми відкатаємо

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git checkout File1.txt

Updated 1 path from the index

. . . перевіряємо, доданий рядок повинен видалитися

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ cat File1.txt

testing line

untesting file

commit test

456

7897.

Add Line

**2. Відкат в декількох файлах одразу**

Додамо рядки у всі створені файли . . .

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ echo "line" >> conflict

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ echo "line" >> File1.txt

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ echo "line" >> new.file

. . . перевіримо чи додалася інфа

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ cat conflict

line

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ cat File1.txt

testing line

untesting file

commit test

456

7897.

Add Line

line

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ cat new.file

new line

line

. . . тепер щоб зробити відкат одразу у трьох файлах виконаємо команду git checkout, але замість назви файлу поставимо крапку

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git checkout .

Updated 3 paths from the index

**Тимчасове сховище**

Також ми можемо додавати зміни у тимчасове сховище, тимчасово скривати

**1. Додамо рядок**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ echo "line" >> conflict

**2. Перевіримо, рядок додався**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ cat conflict

line

**3. Тепер перенесемо цю зміну у сховище**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git stash

warning: in the working copy of 'conflict', LF will be replaced by CRLF the ne

xt time Git touches it

Saved working directory and index state WIP on main: 17b0874 test

**4. Перевіримо, що після перенесення рядка у сховища йогоне видно**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ cat conflict

Рядок зник

**5. Але ми завжди можемо повернути зміниу зі сховища, використавши команду**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git stash pop

On branch main

Your branch is up to date with 'origin/main'.

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: conflict

no changes added to commit (use "git add" and/or "git commit -a")

Dropped refs/stash@{0} (2584a85504af5131132103d8be819fe205919d82)

**6. Перевіряємо, рядок повинен з'явитися**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ cat conflict

line

**7. Для того щоб назавжди видалити інформацію зі сховища, використовуємо команду**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git stash clear

8. Перевіряємо чи видалилася інформація зі сховища

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git stash pop

No stash entries found.

**Робота з додатковими гілками (Branch)**

**1. Для створення нової гілки використовуємо команду**

git branch назва нової гілки

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git branch first\_branch

**2. Для перевірки, чи створилася нова гілка використовуємо команду**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git branch

first\_branch

\* main

. . .завдяки цієї команді ми бачимо, що заразу нас є основна гілка \* main, і одна додаткова

first\_branch. \* - показано, що зараз ми знаходимося на основній гілці

**3. Для того щоб нам перейти для роботи у додаткову гілку (Branch), використовуємо команду**

git checkout назва створеної, або необхідної гілки

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git checkout first\_branch

Switched to branch 'first\_branch'

M conflict

**4. Тепер треба перевірити чи ми дійсно перейшли на гілку, виконуємо команду**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (first\_branch)

$ git branch

\* first\_branch

main

. . . тепер \* перейшла на назву нашої додаткової гілки і це означає, що ми перейшли в неї

**5. Попрацюємо з вже створеним файлом додамо рядок**

echo "інфа яка буде у новому рядку" >> назва файлу в який ми хочемо додоти інфу

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (first\_branch)

$ echo "new branch line" >> File1.txt

**6. Перевіримо**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (first\_branch)

$ cat File1.txt

testing line

untesting file

commit test

456

7897.

Add Line

new branch line

. . . також щоб обширніше бачити зміни можна використати команду

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (first\_branch)

$ git diff

warning: in the working copy of 'File1.txt', LF will be replaced by CRLF the nex

t time Git touches it

diff --git a/File1.txt b/File1.txt

index aa316d8..c1d036e 100644

--- a/File1.txt

+++ b/File1.txt

@@ -9,3 +9,4 @@ commit test

7897.

Add Line

+new branch line

diff --git a/conflict b/conflict

index 8b13789..f465726 100644

--- a/conflict

+++ b/conflict

@@ -1 +1,2 @@

+line

. . . перевіримо в якому статусі знаходится наша

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (first\_branch)

$ git status

On branch first\_branch

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: File1.txt

modified: conflict

no changes added to commit (use "git add" and/or "git commit -a")

**7. Додамо нашу змінену інформацію**

git commit -am "короткий опис зміненої інформації"

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (first\_branch)

$ git commit -am "new branch commit"

warning: in the working copy of 'File1.txt', LF will be replaced by CRLF the nex

t time Git touches it

[first\_branch 4a0d93e] new branch commit

2 files changed, 2 insertions(+)

[first\_branch 4a0d93e] – цей напис нам показує в якій гілці була змінена інформація

**8. Перенесемо змінену іформацію з локального репозиторія на віддаленний**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (first\_branch)

$ git push

$ git push

fatal: The current branch first\_branch has no upstream branch.

To push the current branch and set the remote as upstream, use

git push --set-upstream origin first\_branch

To have this happen automatically for branches without a tracking

upstream, see 'push.autoSetupRemote' in 'git help config'.

. . . як ми бачимо що нам видає помилку, що на віддаленому репозиторії у нас немає гілки з якою ми можемо з'єднатися, тому нам треба це зробити

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (first\_branch)

$ git push –u origin first\_branch

Enumerating objects: 7, done.

Counting objects: 100% (7/7), done.

Delta compression using up to 4 threads

Compressing objects: 100% (3/3), done.

Writing objects: 100% (4/4), 390 bytes | 390.00 KiB/s, done.

Total 4 (delta 0), reused 0 (delta 0), pack-reused 0

remote:

remote: Create a pull request for 'first\_branch' on GitHub by visiting:

remote: https://github.com/Svetochek30/Test/pull/new/first\_branch

remote:

To https://github.com/Svetochek30/Test.git

\* [new branch] first\_branch -> first\_branch

branch 'first\_branch' set up to track 'origin/first\_branch'.

**9. Перевіримо як визлядає наша інформацію після змін і перенесення у віддаленний репозиторій**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (first\_branch)

$ cat File1.txt

testing line

untesting file

commit test

456

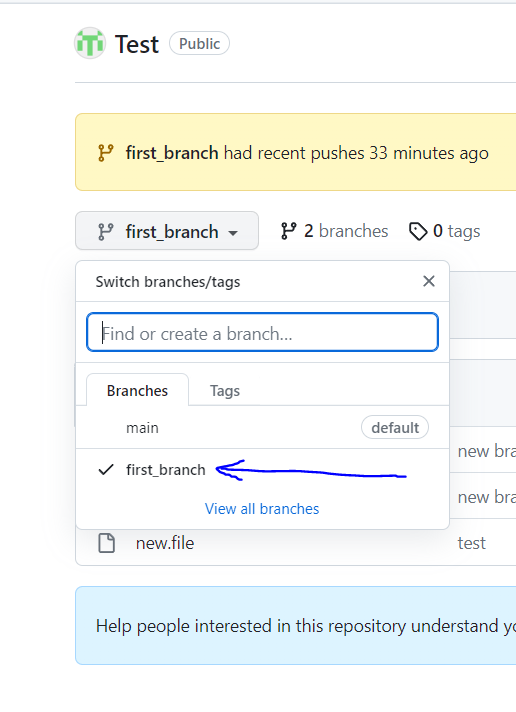
7897.

Add Line

new branch line

. . . все добре зміни зберілися

**10. Перевиримо чи відбулися зміни на віддаленному репозиторії**



. . . так з'явилася гілка

**Додавання додакової гілки в основну**

**1. Тепер уявимо, що на потрібно добавити додаткову гілку (Branch) у основну гілку змержити, але спочатку на треба повернутися в основну гілку, для цього використаємо команду**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (first\_branch)

$ git checkout -

Switched to branch 'main'

Your branch is up to date with 'origin/main'.

**2. Перенесемо додаткову гілку в основну**

git merge назва гілки яку будумо переносити

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git merge first\_branch

Updating 17b0874..4a0d93e

Fast-forward

File1.txt | 1 +

conflict | 1 +

2 files changed, 2 insertions(+)

. . .ще не все. . .

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git push

Total 0 (delta 0), reused 0 (delta 0), pack-reused 0

To https://github.com/Svetochek30/Test.git

17b0874..4a0d93e main -> main

. . . тепер все, перевіримо

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

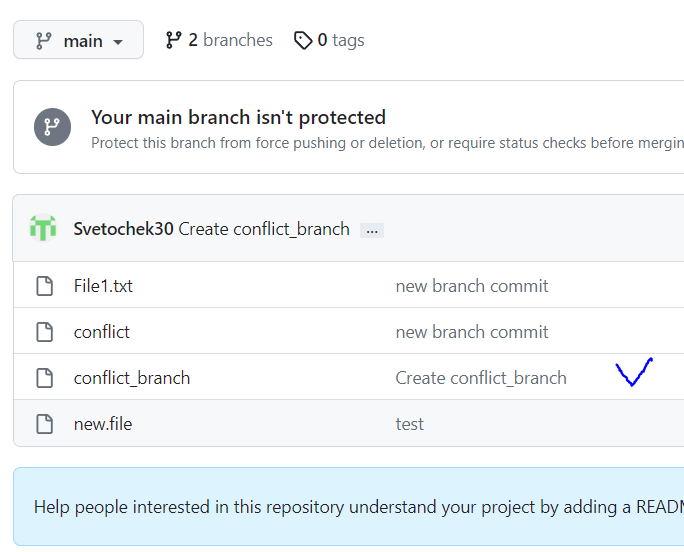
nothing to commit, working tree clean

**Конфлікти при об'єднанні гілок**

Іноді не все проходить просто, може виникнути конфлікт, як його вирішити

**Конфлікт 1**. Якщо ми у віддаленому репозиторії додали новий файл, а у локальному змінили поточний файл.

**1. Додамо у віддалений репозиторій навий файл**



**2. Змінемо інформацію у вже інуючому файлі на локальному репозиторії**

**3. Додамо нову гілку**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git branch second\_branch

**4. Виконаємо команду для переходу для роботи у додаткову гілку (Branch)**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git checkout

M File1.txt

Your branch is up to date with 'origin/main'.

**5. Перевіримо чи створилася нова гілка**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git branch

first\_branch

\* main

second\_branch

**6. Виконаємо команду щоб перейти для роботи на нову гілку**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git checkout second\_branch

Switched to branch 'second\_branch'

M File1.txt

**7. У новій гілці створимо файлик з новим рядком**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (second\_branch)

$ echo "test\_line" >> second.file

**8. Перевіримо**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (second\_branch)

$ git status

On branch second\_branch

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: File1.txt

Untracked files:

(use "git add <file>..." to include in what will be committed)

second.file

no changes added to commit (use "git add" and/or "git commit -a")

**9. Додамо зміни**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (second\_branch)

$ git add second.file

warning: in the working copy of 'second.file', LF will be replaced by CRLF the n

ext time Git touches it

**10. Проведемо зміни**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (second\_branch)

$ git commit -am "add second.file"

[second\_branch 06ceb53] add second.file

2 files changed, 2 insertions(+), 6 deletions(-)

create mode 100644 second.file

**11. Запушимо**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (second\_branch)

$ git push -u origin second\_branch

Enumerating objects: 6, done.

Counting objects: 100% (6/6), done.

Delta compression using up to 4 threads

Compressing objects: 100% (3/3), done.

Writing objects: 100% (4/4), 429 bytes | 429.00 KiB/s, done.

Total 4 (delta 0), reused 0 (delta 0), pack-reused 0

remote:

remote: Create a pull request for 'second\_branch' on GitHub by visiting:

remote: https://github.com/Svetochek30/Test/pull/new/second\_branch

remote:

To https://github.com/Svetochek30/Test.git

\* [new branch] second\_branch -> second\_branch

branch 'second\_branch' set up to track 'origin/second\_branch'.

**12. Спробуємо провести мердж у майстер гілку**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (second\_branch)

$ git checkout main

Switched to branch 'main'

Your branch is up to date with 'origin/main'.

**13. Змержемо гілку**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git merge second\_branch

Updating 4a0d93e..06ceb53

Fast-forward

File1.txt | 7 +------

second.file | 1 +

2 files changed, 2 insertions(+), 6 deletions(-)

create mode 100644 second.file

**14. Спробуемо запушити**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git push

To https://github.com/Svetochek30/Test.git

! [rejected] main -> main (fetch first)

error: failed to push some refs to 'https://github.com/Svetochek30/Test.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.

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

. . . виникла помилка, тому що в нас не співпадає інформація з віддаленим і локальним репозиторієм

**15. Це вирішується декількома командами**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git pull

remote: Enumerating objects: 4, done.

remote: Counting objects: 100% (4/4), done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (3/3), 741 bytes | 92.00 KiB/s, done.

From https://github.com/Svetochek30/Test

4a0d93e..c3a1579 main -> origin/main

Merge made by the 'ort' strategy.

conflict\_branch | 1 +

1 file changed, 1 insertion(+)

create mode 100644 conflict\_branch

. . . тепер пушимо

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git push

Enumerating objects: 4, done.

Counting objects: 100% (4/4), done.

Delta compression using up to 4 threads

Compressing objects: 100% (2/2), done.

Writing objects: 100% (2/2), 330 bytes | 330.00 KiB/s, done.

Total 2 (delta 1), reused 0 (delta 0), pack-reused 0

remote: Resolving deltas: 100% (1/1), completed with 1 local object.

To https://github.com/Svetochek30/Test.git

c3a1579..08b7bdc main -> main

. . . тепер все повинно виправитися, перевіряємо статус

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

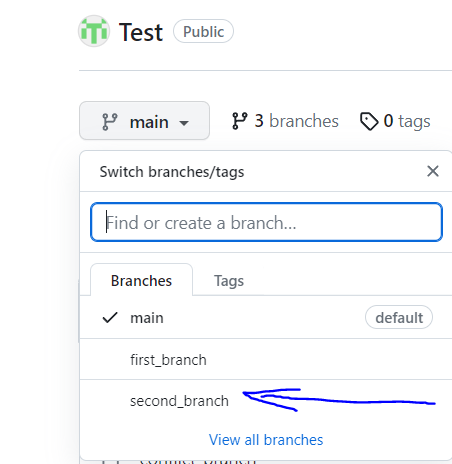
$ git status

On branch main

Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

на локальному репозиторіії все добре, перевіряємо віддалений, щобз'явилася нова гілка



**Конфлікт 2.** Коли відбуваються зміни в одному і томуж файлі, в одному і тому ж рядку

**1. Створемо нову гілку**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git branch new\_file\_edit

**2. Перейдімо у цю гілку**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git checkout new\_file\_edit

Switched to branch 'new\_file\_edit'

**3. Змінемо інформацію у файлі на віддаленому репозиторії**

**4. Змінемо інформацію у тому ж файлі у тому ж рядку але внесемо іншу інформацію на локальному репозиторії**

**5. Проведемо зміни комміт**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (new\_file\_edit)

$ git commit -am "edit2023"

[new\_file\_edit b4d80ab] edit2023

1 file changed, 1 insertion(+), 1 deletion(-)

**6. Запушимо**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (new\_file\_edit)

$ git push -u origin new\_file\_edit

Enumerating objects: 12, done.

Counting objects: 100% (11/11), done.

Delta compression using up to 4 threads

Compressing objects: 100% (6/6), done.

Writing objects: 100% (8/8), 1.34 KiB | 1.34 MiB/s, done.

Total 8 (delta 1), reused 0 (delta 0), pack-reused 0

remote: Resolving deltas: 100% (1/1), done.

remote:

remote: Create a pull request for 'new\_file\_edit' on GitHub by visiting:

remote: https://github.com/Svetochek30/Test/pull/new/new\_file\_edit

remote:

To https://github.com/Svetochek30/Test.git

\* [new branch] new\_file\_edit -> new\_file\_edit

branch 'new\_file\_edit' set up to track 'origin/new\_file\_edit'.

**7. Тепер треба перенести (змержити) додаткову гілку в основну**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (new\_file\_edit)

$ git checkout main

Switched to branch 'main'

Your branch is up to date with 'origin/main'.

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git merge new\_file\_edit

Updating 08b7bdc..b4d80ab

Fast-forward

new.file | 2 +-

1 file changed, 1 insertion(+), 1 deletion(-)

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git push

To https://github.com/Svetochek30/Test.git

! [rejected] main -> main (fetch first)

error: failed to push some refs to 'https://github.com/Svetochek30/Test.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.

. . . видає помилку бо у нас є змінена інформація в обох репозиторіях

Виправляємо!!!

**1. виконуємо команду, щоб нам показало що саме викликає конфлікт**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git pull

remote: Enumerating objects: 5, done.

remote: Counting objects: 100% (5/5), done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (3/3), 655 bytes | 40.00 KiB/s, done.

From https://github.com/Svetochek30/Test

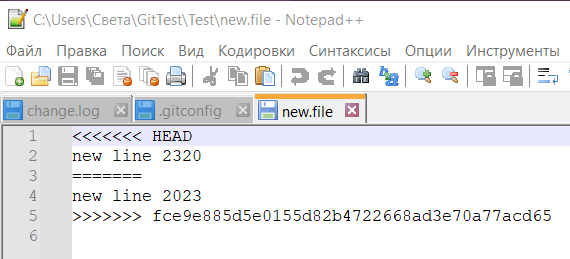
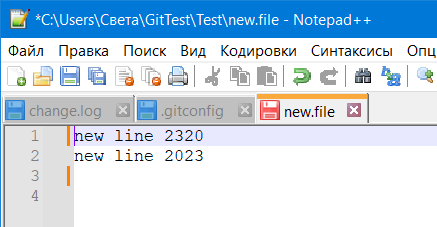
08b7bdc..fce9e88 main -> origin/main

Auto-merging new.file

CONFLICT (content): Merge conflict in new.file

Automatic merge failed; fix conflicts and then commit the result.

**2. Заходимо на локальному комп'ютері у файл де відбулися зміни, і бачимо 2 варіати виправлень, ми можемо залишити обидва, аба залишити якийсь один, а не потрібну іформацію видаляємо**

**3. Перевіряємо чи вірна інформація залишилася у файлі**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main|MERGING)

$ cat new.file

new line 2320

new line 2023

**4. Виправляємо**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main|MERGING)

$ git add new.file

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main|MERGING)

$ git commit -am "changes are approved"

[main c20ebed] changes are approved

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git push

Enumerating objects: 7, done.

Counting objects: 100% (7/7), done.

Delta compression using up to 4 threads

Compressing objects: 100% (2/2), done.

Writing objects: 100% (3/3), 313 bytes | 313.00 KiB/s, done.

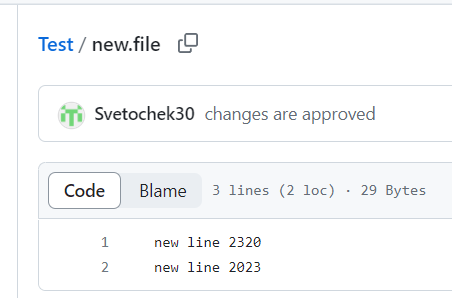
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0

remote: Resolving deltas: 100% (1/1), completed with 1 local object.

To https://github.com/Svetochek30/Test.git

fce9e88..c20ebed main -> main

**5. Перевіряємо на віддаленому репозиторії**



. . . все добре, все оновилося

**Перейменування гілок**

**Варіант 1**. Коли є тільки гілка, а файлів ще не має

**1. Створемо гілку**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git branch first\_name

**2. Переіменовуємо виконавши команду**

git branch -m старе ім'я гілки нове ім'я гілки

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git branch -m first\_name second\_name

**3. Перевіряємо**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git branch

first\_branch

\* main

new\_file\_edit

second\_branch

second\_name

**Варіант 2.** Коли у гілці вже є файли

**1. Перейдемо у гілку**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git checkout second\_name

Switched to branch 'second\_name'

**2. Створемо у гілці файл**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (second\_name)

$ touch second\_name

**3. Додамо файл до гілки**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (second\_name)

$ git add second\_name

**4. Проведемо наші зміни (комміт)**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (second\_name)

$ git commit -am "add second.name"

[second\_name b5689ba] add second.name

1 file changed, 0 insertions(+), 0 deletions(-)

create mode 100644 second\_name

**5. Запушимо**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (second\_name)

$ git push -u origin second\_name

Enumerating objects: 4, done.

Counting objects: 100% (4/4), done.

Delta compression using up to 4 threads

Compressing objects: 100% (2/2), done.

Writing objects: 100% (3/3), 275 bytes | 275.00 KiB/s, done.

Total 3 (delta 1), reused 0 (delta 0), pack-reused 0

remote: Resolving deltas: 100% (1/1), completed with 1 local object.

remote:

remote: Create a pull request for 'second\_name' on GitHub by visiting:

remote: https://github.com/Svetochek30/Test/pull/new/second\_name

remote:

To https://github.com/Svetochek30/Test.git

\* [new branch] second\_name -> second\_name

branch 'second\_name' set up to track 'origin/second\_name'.

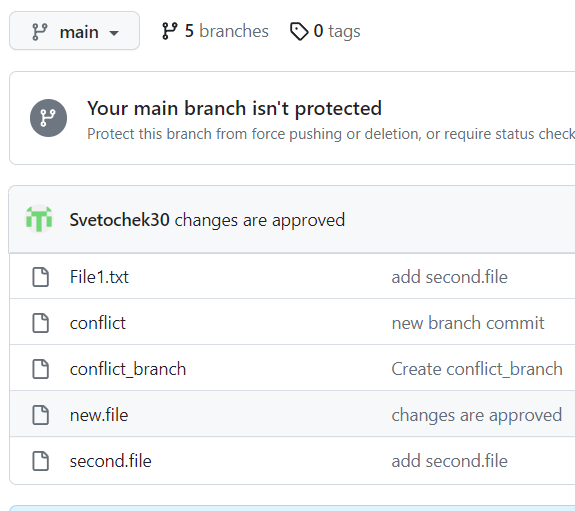
**6. Пробуємо переіменувати гілку**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (second\_name)

$ git branch -m second\_name first\_name

. . . на локальному репозиторіії все пройшло

**7. Дивимось чи пройшли зміни на віддаленому репозиторії**



. . . Нічого не відбулося, залишилося старе ім'я файлу

**8. Щоб це виправити, нам потрібно на локальному репозиторії видалити гілку зі старим ім'ям та запушити з новим ім'ям**

. . .видаляємо. . .

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (first\_name)

$ git push origin :second\_name

To https://github.com/Svetochek30/Test.git

- [deleted] second\_name

. . . пушимо нове ім'я. . .

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (first\_name)

$ git push -u origin first\_name

Enumerating objects: 4, done.

Counting objects: 100% (4/4), done.

Delta compression using up to 4 threads

Compressing objects: 100% (2/2), done.

Writing objects: 100% (3/3), 275 bytes | 275.00 KiB/s, done.

Total 3 (delta 1), reused 0 (delta 0), pack-reused 0

remote: Resolving deltas: 100% (1/1), completed with 1 local object.

remote:

remote: Create a pull request for 'first\_name' on GitHub by visiting:

remote: https://github.com/Svetochek30/Test/pull/new/first\_name

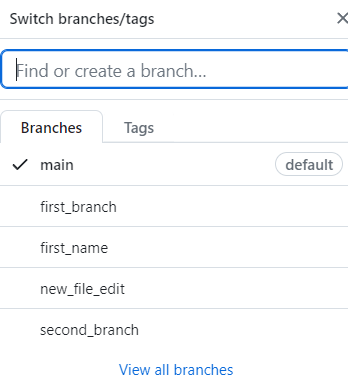
remote:

To https://github.com/Svetochek30/Test.git

\* [new branch] first\_name -> first\_name

branch 'first\_name' set up to track 'origin/first\_name'.

. . . перевіряємо зміни на віддаленому репозиторії



**Видалення гілки з локального репозиторія**

**1. Переходимо на головну майтер гілку**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (first\_name)

$ git checkout main

Switched to branch 'main'

Your branch is up to date with 'origin/main'.

**2. Вводимо команду для видалення**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git branch -d second\_branch

Deleted branch second\_branch (was 06ceb53).

. . . все вона видалена

**3. Якщо ми помилково її видалили і нам треба її повернути**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git fetch origin second\_branch

From https://github.com/Svetochek30/Test

\* branch second\_branch -> FETCH\_HEAD

**4. Переходимо в повернуту гілку**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (main)

$ git checkout second\_branch

Switched to a new branch 'second\_branch'

branch 'second\_branch' set up to track 'origin/second\_branch'.

**5. Перевіряємо**

Света@DESKTOP-2G1VQM4 MINGW64 ~/GitTest/Test (second\_branch)

$ git branch

first\_branch

first\_name

main

new\_file\_edit

\* second\_branch

. . .так все добре