

# Zastosowania informatyki w pracy grupowej

## Laboratorium 2 - GIT

**Proszę otworzyć okno linii komend po czym:**

1. stworzyć nowy folder i wewnątrz niego foldery *local*, *local2* oraz *remote*,
2. przejść do folderu *local* i założyć w nim nowe repozytorium GIT:  
`git init`
3. przejść do folderu *remote* założyć czyste zdalne repozytorium komendą:  
`git init --bare`
4. wrócić do *local*, stworzyć plik o nazwie `wiersz.txt` i umieścić w nim tekst  
wiersza (polskie znaki nie są konieczne):  
Spotkali się w święto o piątej przed kinem  
Miejscowa idiotka z tutejszym kretynem.  
  
Tutejsza idiotka - rzekł kretyn miejscowy -  
Czy pragniesz pójść ze mną na film przebojowy?
5. sprawdzić stan folderu komendą  
`git status`
6. dodać nowo stworzony plik do poczekalni (ang. Staging area) używając funkcji  
*add gita*
7. sprawdzić ponownie stan folderu odpowiednią komendą,
8. co się zmieniło?
9. następnie wykonać commit z opisem: „*pierwsze 2 zwrotki*”:  
`git commit`  
*W jaki inny sposób można wykonać commit podając opis bezpośrednio jako parametr?*
10. sprawdzić raz jeszcze stan folderu komendą:  
`git status`
11. ustawić jako zdalne zdalne repozytorium uprzednio stworzony i zainicjalizowany folder *remote*:  
`git remote add origin ../remote`
12. sprawdzić aktualnie ustawione zdalne repozytoria:  
`git remote -v`
13. wysłać dokonane zmiany w repozytorium lokalnym do zdalnego:  
`git push -u origin master`

14. zadeklarować GITowi, aby nie zawierał plików `.doc` w repozytorium i zapisać tą zmianę z opisem „*ignorowanie .doc*”:

```
echo "*.doc" > .gitignore
```

Dalsze wymagane komendy proszę wymyślić na podstawie wcześniejszych punktów.

15. stworzyć plik `smieci.doc` o treści: „*nie ma tu nic ciekawego*”,

16. sprawdzić raz jeszcze stan folderu odpowiednią komendą i potwierdzić czy punkt 14 wykonano poprawnie,

17. zapisać log zmian w repozytorium w pliku `log.txt` po czym dodać go do przechowalni i zapisać z opisem „*log*”:

```
git log > log.txt
```

Dalsze wymagane komendy proszę wymyślić na podstawie wcześniejszych punktów.

18. uaktualnić zdalne repozytorium,

19. w pliku `wiersz.txt` zamienić słowa „*kretyn*” na innego dowolnie wybranego bohatera,

20. wygenerować plik o nazwie `roznica.txt` zawierający różnice w stosunku do oryginału pliku:

```
git diff > roznica.txt
```

21. przejść do katalogu `local2` i sklonować do niego zawartość zdalnego repozytorium `remote`:

```
git clone ../remote
```

22. do pliku `wiersz.txt` ( w katalogu `local2/remote` ) dodać kolejne zwrotki:

Miejscowa kretynka odrzekła: - Z ochotą,  
Albowiem cię kocham, tutejszy idioto.

Więc kretyn miejscowy uśmiechnął się słodko  
I poszedł do kina z tutejszą idiotką.

23. zapisać zmiany w lokalnym repozytorium (`local2/remote`) z opisem „*3 i 4 zwrotka*” po czym wysłać je do zdalnego.

Jak możemy w tym wypadku zrobić to tylko dwoma komendami?

24. wrócić do folderu `local` i zapisać do repozytorium lokalnego (folder `local`) tylko plik `wiersz.txt` z opisem „*nowy bohater*” (`roznica.txt` nie dodajemy),

25. założyć nową gałąź „*moja\_wersja*”:

```
git branch moja_wersja
```

26. sprawdzić czy operacja zakończyła się sukcesem:

```
git branch
```

27. pozostając nadal w gałęzi *master*, wysłać zmiany do *zdalnego repozytorium*.

W *wiersz.txt* ma pozostać pierwsza zwrotka z „nowym bohaterem” oraz druga zwrotka oryginalna. Opis commita proszę zostawić domyślny.

Standardowo należałoby wykonać:

```
git push
```

Dlaczego w ten sposób się to nie udaje?

Rozwiązaniem problemu będzie:

```
git fetch
```

```
git merge
```

```
git push
```

Jaką jedną komendą możemy zastąpić dwie pierwsze?

28. przejść do utworzonej w punkcie 25 gałęzi „*moja\_wersja*”:

```
git checkout moja_wersja
```

29. sprawdzić co teraz znajduje się w pliku *wiersz.txt* i uzasadnić skąd taka treść,

30. na końcu pliku dopisać własną wymyśloną kolejną zwrotkę,

31. dodać plik *roznica.txt* oraz *wiersz.txt* w nowej wersji do repozytorium w aktualnej gałęzi z opisem „*moja zwrotka*” i wysłać repozytorium zdalnego,

32. wrócić do gałęzi *master*,

33. zwrócić uwagę jakiego pliku brakuje. Dlaczego?

34. cofnąć merge pliku *wiersz.txt* i wrócić do jego stanu zawierającego dwie zwrotki z oryginalnym bohaterem. Do tego celu używamy komendy

```
git revert 1234
```

gdzie 1234 jest identyfikatorem (skróttem SHA danego commita). Aby uzyskać numer odpowiedniego commita należy użyć komendy

```
git log
```

w wypadku cofania merga (który jest połączeniem dwóch różnych wersji), należy podać do której wersji chcemy powrócić:

```
git revert -m 1 1234
```

lub

```
git revert -m 2 1234
```

Która opcja będzie poprawna w tym wypadku?

35. dodać pozostałe zwrotki:

Na miłym macaniu spłynęła godzinka  
I była szczęśliwa miejscowa kretynka.

Aż wreszcie szepnęła: - Kretynie tutejszy!  
Ten film, mam wrażenie, jest coraz nudniejszy.

Więc poszli na sznycel, na melbę, na winko,

Miejscowy idiota z tutejszą kretynką.

Następnie się zwarli w uścisku zmysłowym  
Tutejsza idiotka z kretynem miejscowym.

W ten sposób dorobią się córki lub syna:  
Idioty, idiotki, kretynki, kretyna,

By znowu się mogli spotykać przed kinem  
Tutejsza idiotka z miejscowym kretynem.

zapisać je i wysłać do *repozytorium zdalnego*.

36. Proszę przejść do folderu *local2/remote* i uaktualnić to repozytorium do najnowszej wersji.