Sprawozdanie (Lista 2)

Hanna Ramanouskaya, numer indeksu: 272395

**Zadanie1:**

# Zadanie: Doprowadzenie do konfliktu w SVN i ręczne rozwiązanie konfliktu

* **Krok 1: Utworzenie pliku w pierwszej kopii roboczej i zatwierdzenie zmian**

# Przejście do pierwszej kopii roboczej

$ svn checkout https://repo.cs.pwr.edu.pl/mk1-272395/SP/lab2/zadanie1 kopia1

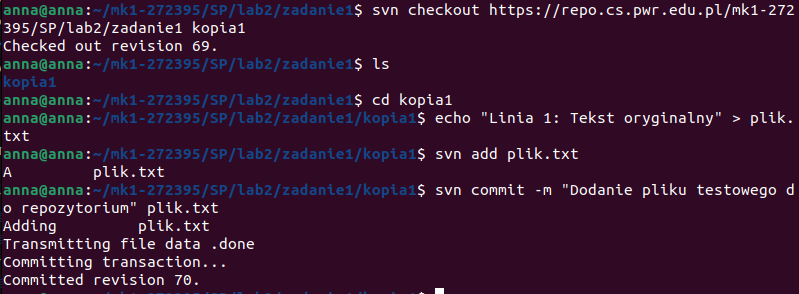
$ cd kopia1

# Utworzenie pliku testowego i dodanie go do repozytorium

$ echo "Linia 1: Tekst oryginalny" > plik.txt

$ svn add plik.txt

$ svn commit -m "Dodanie pliku testowego do repozytorium" plik.txt



* **Krok 2: Wprowadzenie zmian w pliku w obu kopiach roboczych**

# Przejście do drugiej kopii roboczej

$ cd ..

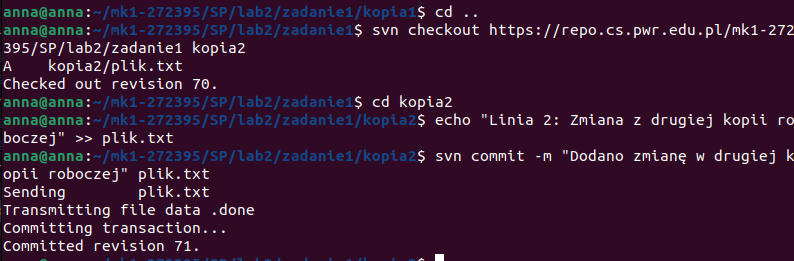
$ svn checkout <https://repo.cs.pwr.edu.pl/mk1-272395/SP/lab2/zadanie1> kopia2

$ cd kopia2

# Wprowadzenie zmiany w drugiej kopii roboczej

$ echo "Linia 2: Zmiana z drugiej kopii roboczej" >> plik.txt

$ svn commit -m "Dodano zmianę w drugiej kopii roboczej" plik.txt

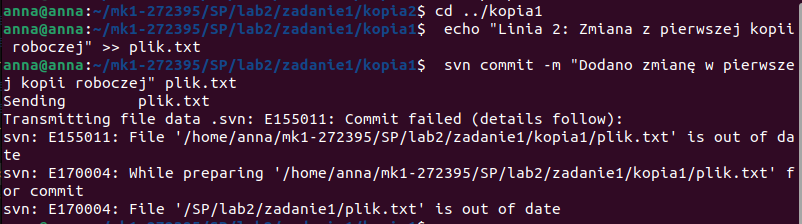


# Przejście do pierwszej kopii roboczej i wprowadzenie innej zmiany

$ cd ../kopia1

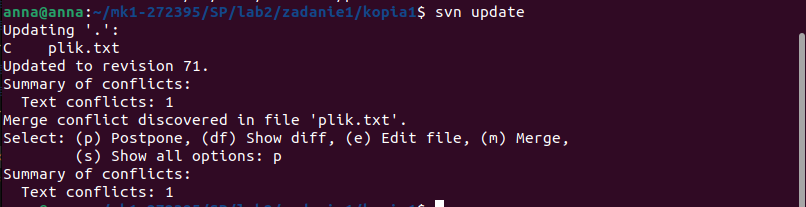
$ echo "Linia 2: Zmiana z pierwszej kopii roboczej" >> plik.txt

$ svn commit -m "Dodano zmianę w pierwszej kopii roboczej" plik.txt



* **Krok 3: Aktualizacja pierwszej kopii roboczej, aby doprowadzić do konfliktu**

$ svn update



# Wybieramy opcję "postpone" (p), aby odłożyć rozwiązanie konfliktu na później

# Sprawdzenie statusu plików i plików pomocniczych

$ ls



* **plik.txt**: Zawiera znaczniki konfliktu, które wskazują różnice między wersjami.
* **plik.txt.mine**: Twoja lokalna wersja pliku, przed próbą aktualizacji.
* **plik.txt.r70**: Wersja pliku sprzed konfliktu, odpowiadająca starszej rewizji.
* **plik.txt.r71**: Wersja pliku, która spowodowała konflikt (najnowsza wersja z repozytorium).

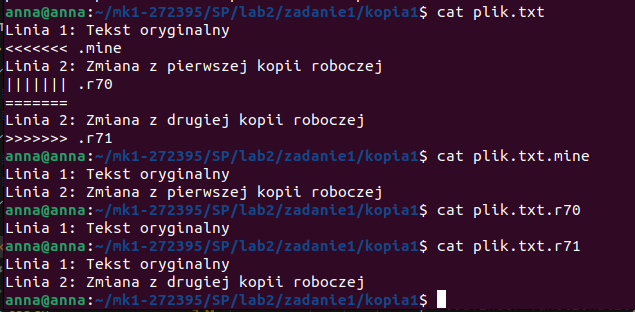
# Wyświetlenie zawartości plików zaangażowanych w konflikt

$ cat plik.txt

$ cat plik.txt.mine

$ cat plik.txt.r70

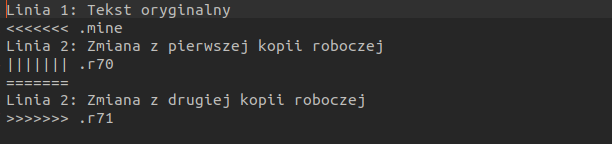
$ cat plik.txt.r71



* **Krok 4: Ręczne rozwiązanie konfliktu**

# Edytujemy plik plik.txt, aby rozwiązać konflikt, np. łącząc obie wersje zmian

# Otwórzamy plik plik.txt w edytorze tekstowym i usuwamy znaczniki konfliktu <<<<<<<, | | | | | | | | |, i >>>>>>>, wybierając odpowiednią wersję lub łącząc obie linie.



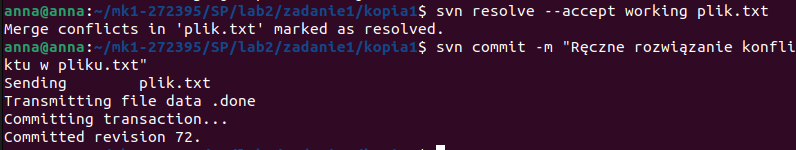
#Po usunięciu:



# Zatwierdzenie rozwiązania konfliktu

$ svn resolve --accept working plik.txt

$ svn commit -m "Ręczne rozwiązanie konfliktu w pliku.txt"

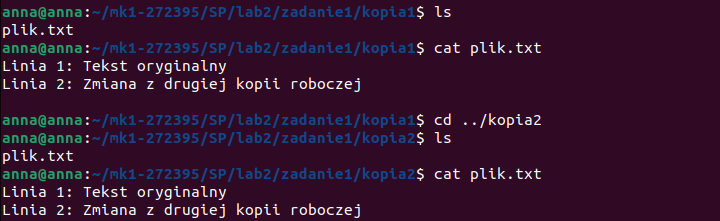


# Sprawdzenie, że pliki pomocnicze zniknęły

$ ls

# Wyświetlenie ostatecznej zawartości pliku po rozwiązaniu konfliktu

$ cat plik.txt

svn log -v -r 70:72 /home/anna/mk1-272395/SP/lab2/zadanie1:$



**Zadanie 2:**

# Zadanie: Utworzenie poddrzewa katalogów, kopiowanie fragmentów podkatalogów z ustawieniem różnych parametrów głębokości. Demonstracja wykorzystania "sparse directories" w SVN.

* **Krok 1: Eksportowanie poddrzewa katalogów**

# Eksportujemy poddrzewo katalogów do cd zadanie2

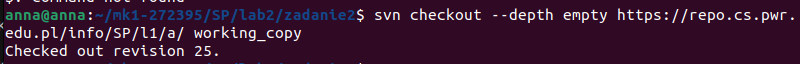
$ svn export <https://repo.cs.pwr.edu.pl/info/SP/l1/a/>



* **Krok 2: Utworzenie nowej kopii roboczej w katalogu "working\_copy"**

# Tworzymy nową kopię roboczą w katalogu "working\_copy"

$ svn checkout --depth empty https://repo.cs.pwr.edu.pl/info/SP/l1/a/ working\_copy

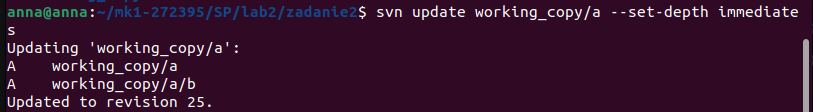


#Polecenie --depth empty tworzy "płytką" kopię roboczą – bez zawartości wewnętrznych katalogów. To pozwala na późniejsze dołączanie tylko tych elementów, które są potrzebne

* **Krok 4: Pobranie katalogu `a` na głębokość "immediates"**

# Pobieramy katalog `a` na głębokość "immediates" (tylko bezpośrednie pliki i podkatalogi)

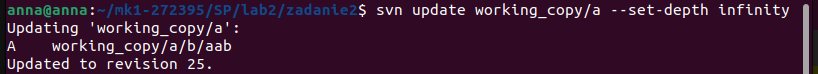
$ svn update working\_copy/a --set-depth immediates



* **Krok 5: Pobranie katalogu `a` na głębokość "infinity"**

# Pobieramy katalog `a` na głębokość "infinity" (wszystkie pliki i podkatalogi)

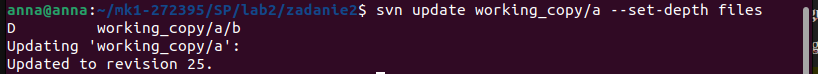
$ svn update working\_copy/a --set-depth infinity



* **Krok 6: Pobranie katalogu `` na głębokość "files"**

# Pobieramy katalog `a` na głębokość "files" (tylko pliki, bez podkatalogów)

$ svn update working\_copy/a --set-depth files



Polecenie svn update working\_copy/subdir1 --set-depth immediates

Cel: Pobranie katalogu subdir1 na głębokość "immediates".

Efekt: Pobieramy tylko bezpośrednio zawarte w subdir1 pliki i podkatalogi, ale bez ich zawartości (czyli nie zagłębiamy się w ich wnętrze). W efekcie widzimy tylko pliki i puste podkatalogi na pierwszym poziomie.

Polecenie svn update working\_copy/subdir2 --set-depth infinity

Cel: Pobranie całego katalogu subdir2 wraz ze wszystkimi jego podkatalogami i plikami, bez ograniczeń głębokości.

Efekt: Pobieramy pełną zawartość subdir2, w tym wszystkie podkatalogi i pliki na dowolnym poziomie zagnieżdżenia.

Polecenie svn update working\_copy/subdir3 --set-depth files

Cel: Pobranie katalogu subdir3 wyłącznie na poziomie plików.

Efekt: Pobieramy tylko pliki znajdujące się bezpośrednio w subdir3, bez podkatalogów.

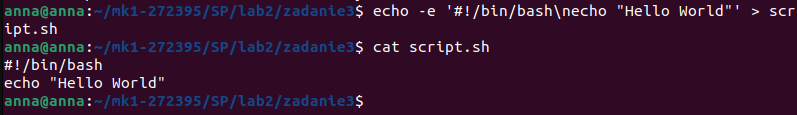
# Wyświetlenie dziennika zmian w podkatalogu zadania

**Zadanie3:**

# Zadanie: Ustawienie i usunięcie własności svn:executable

* **Krok 1: Tworzenie pliku skryptu**

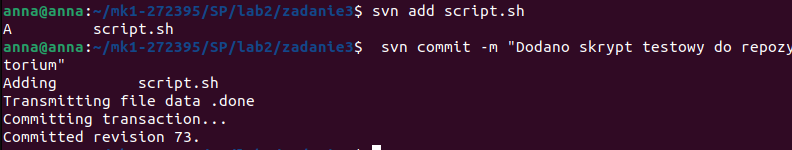
$ echo -e '#!/bin/bash\necho "Hello World"' > script.sh



* **Krok 2: Dodanie skryptu do repozytorium**

$ svn add script.sh

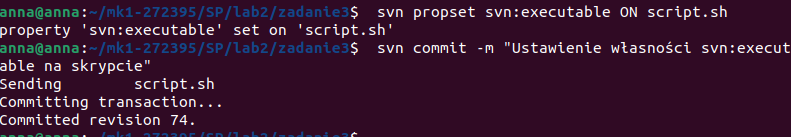
$ svn commit -m "Dodano skrypt testowy do repozytorium"



* **Krok3: Ustawienie własności svn:executable**

$ svn propset svn:executable ON script.sh

$ svn commit -m "Ustawienie własności svn:executable na skrypcie"



* **Krok 4: Testowanie uruchamiania skryptu**

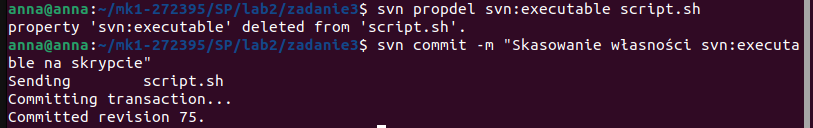
$ ./script.sh



* **Krok 5: Usunięcie własności svn:executable**

$ svn propdel svn:executable script.sh

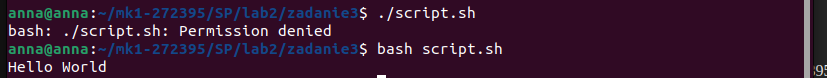
$ svn commit -m "Skasowanie własności svn:executable na skrypcie"



# Ponowne testowanie uruchamiania skryptu bez właściwości executable

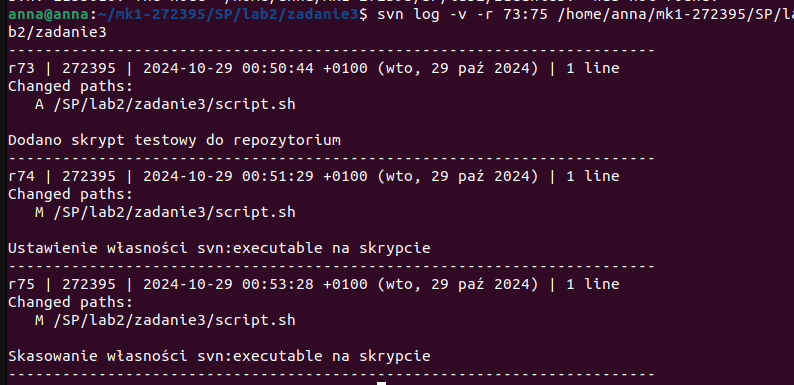
$ ./script.sh

$ bash script.sh



# Ślad działań w repozytorium

$ svn log -v -r 73:75 /home/anna/mk1-272395/SP/lab2/zadanie3



**Zadanie4:**

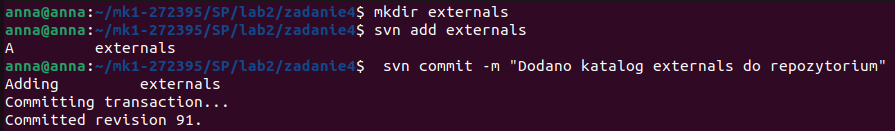
# Zadanie : Ustawienie własności svn:externals na katalogu

* **Krok 1: Przejście do repozytorium i utworzenie katalogu externals**

$ mkdir externals

$ svn add externals

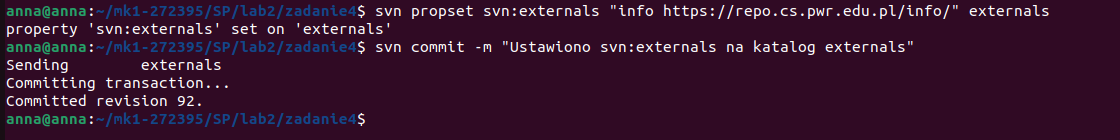
$ svn commit -m "Dodano katalog externals do repozytorium"



* **Krok 2: Ustawienie własności svn:externals na katalogu externals**

$ svn propset svn:externals "info https://repo.cs.pwr.edu.pl/info/" externals

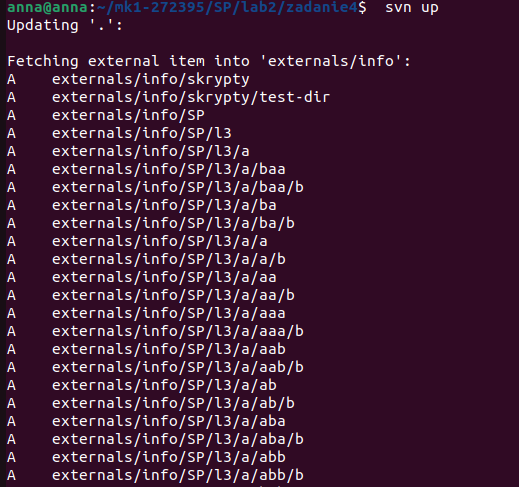
$ svn commit -m "Ustawienie svn:externals na katalogu externals"



* **Krok 3: Sprawdzenie działania svn ckeckout bez opcji --ignore-externals**

# Tworzymy pierwszą kopię roboczą

$ svn checkout https://repo.cs.pwr.edu.pl/mk1-272395/SP/lab2/zadanie4 kopia1



....

A kopia1/externals/info/asciinema/5\_generowanie\_kluczy.cast

A kopia1/externals/info/asciinema/2\_pierwsze\_zadanie.cast

A kopia1/externals/info/repozytorium-svn.txt

A kopia1/externals/info/asciinema/4\_meta-sprawozdanie.cast

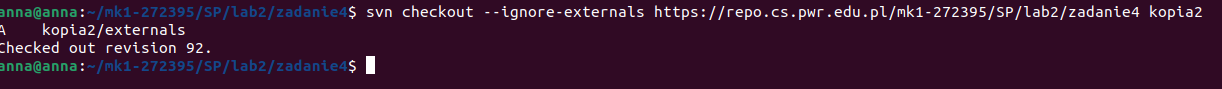
A kopia1/externals/info/asciinema/1\_pierwszy\_checkout.cast

Checked out external at revision 25.

Checked out revision 92.

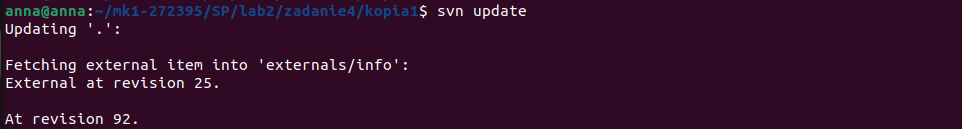
* **Krok 4: Sprawdzenie działania svn ckeckout z opcją --ignore-externals**

$ svn checkout --ignore-externals https://repo.cs.pwr.edu.pl/mk1-272395/SP/lab2/zadanie4 kopia2



* **Krok 5: Testowanie działania svn update bez opcji --ignore-externals (kopia robocza kopia1)**

$ svn update



* **Krok 6: Testowanie działania svn update z opcją --ignore-externals (kopia robocza kopia2)**

$ svn update --ignore-externals



* **svn update bez opcji --ignore-externals**: Pełna aktualizacja, łącznie z zewnętrznym repozytorium externals/info.
* **svn update z opcją --ignore-externals**: Aktualizuje główne repozytorium, ignorując katalog externals/info.
* **svn checkout bez opcji --ignore-externals**: Tworzy pełną kopię repozytorium wraz z zewnętrznym repozytorium externals/info.
* **svn checkout z opcją --ignore-externals**: Tworzy kopię głównego repozytorium bez zewnętrznych katalogów.

**Zadanie 5:**

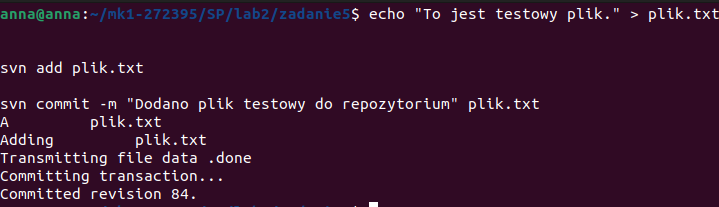
# Zadanie: Testowanie operacji związanych z blokadą plików w SVN

* **Krok 1: Tworzenie przykładowego pliku do testów blokady**

$ echo "To jest testowy plik." > plik.txt

$ svn add plik.txt

$ svn commit -m "Dodano plik testowy do repozytorium" plik.txt



* **Krok 2: Zakładanie blokady na pliku**

$ svn lock plik.txt -m "Blokada do testu"



# Sprawdzenie statusu pliku, aby upewnić się, że blokada jest założona

$ svn status -u



* **Krok 3: Próba usunięcia blokady**

$ svn unlock plik.txt

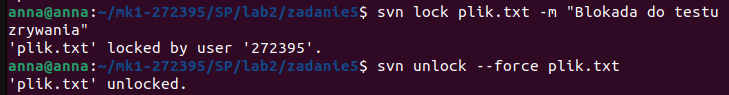


* **Krok 4: Ponowne zakładanie blokady, aby przetestować inne operacje**

$ svn lock plik.txt -m "Blokada do testu zrywania"

# Zrywanie blokady (tzw. break lock) - w celu przejęcia blokady

$ svn unlock --force plik.txt



# Ponowne zakładanie blokady przez jednego użytkownika

$ svn lock plik.txt -m "Blokada do testu podkradania"

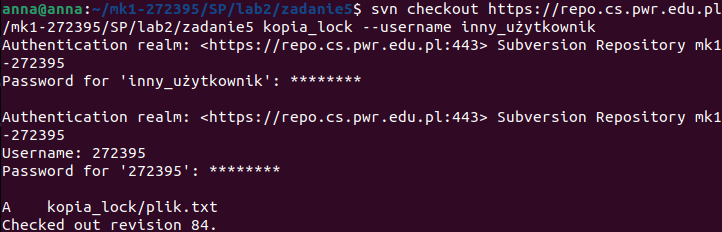
# Podkradanie blokady przez innego użytkownika (symulacja)

# Dla tej operacji potrzebna jest druga kopia repozytorium lub inny użytkownik.

# Zakładamy, że kopiujemy repozytorium jako inny użytkownik lub z innej lokalizacji.

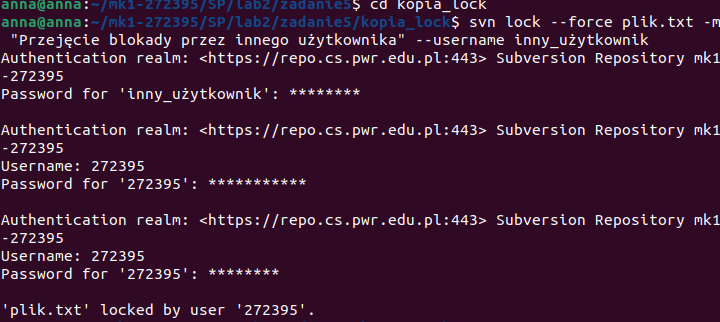
svn checkout https://repo.cs.pwr.edu.pl/mk1-272395/SP/lab2/zadanie5 kopia\_lock --username inny\_użytkownik

$ cd kopia\_lock



# Próba podkradania blokady

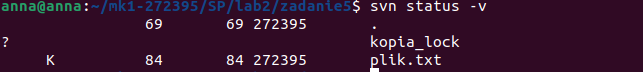
$ svn lock --force plik.txt -m "Przejęcie blokady przez innego użytkownika"



# Sprawdzenie statusu pliku w oryginalnej kopii repozytorium

$ cd ../zadanie5

$ svn status -u



**Zadanie 6**:

#Zadanie : Operacje na katalogach i plikach w SVN

* **Krok 1: Dodanie katalogów a i b z plikami a.txt i b.txt**

$ svn mkdir a

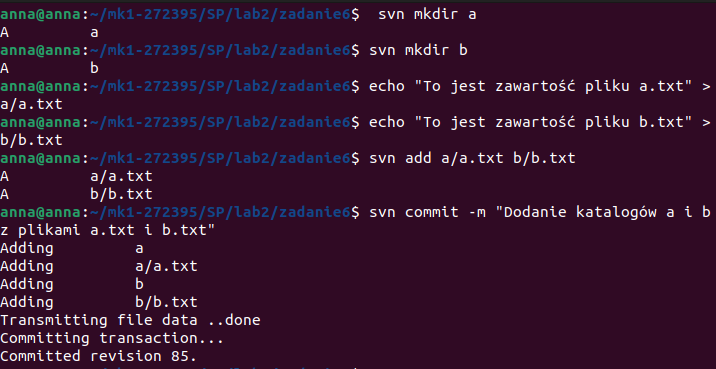
$ svn mkdir b

$ echo "To jest zawartość pliku a.txt" > a/a.txt

$ echo "To jest zawartość pliku b.txt" > b/b.txt

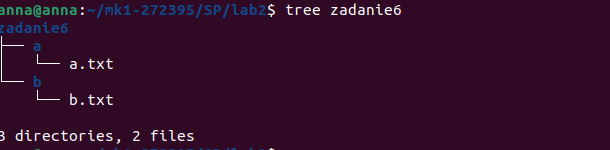
$ svn add a/a.txt b/b.txt

$ svn commit -m "Dodanie katalogów a i b z plikami a.txt i b.txt"



# Struktura katalogów po kroku 1:

$ tree zadanie6



* **Krok 2: Przeniesienie pliku b.txt do katalogu a oraz a.txt do katalogu b**

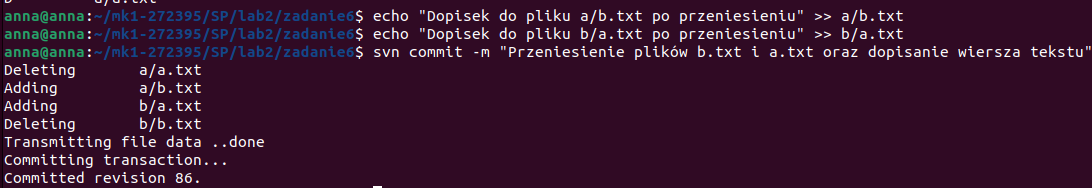
$ svn move b/b.txt a/b.txt

$ svn move a/a.txt b/a.txt

echo "Dopisek do pliku a/b.txt po przeniesieniu" >> a/b.txt

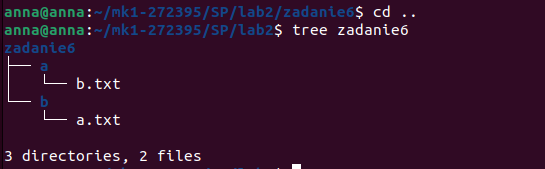
echo "Dopisek do pliku b/a.txt po przeniesieniu" >> b/a.txt

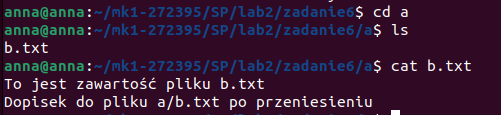
$ svn commit -m "Przeniesienie plików b.txt i a.txt oraz dopisanie wiersza tekstu"

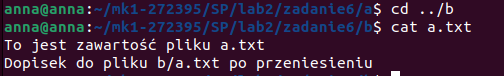


# Struktura katalogów po kroku 2:

$ tree zadanie6





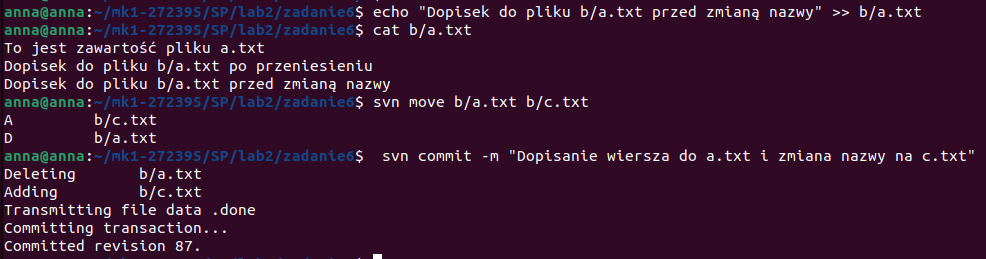


* **Krok 3: Dopisanie wiersza do a.txt i zmiana nazwy na c.txt**

echo "Dopisek do pliku b/a.txt przed zmianą nazwy" >> b/a.txt

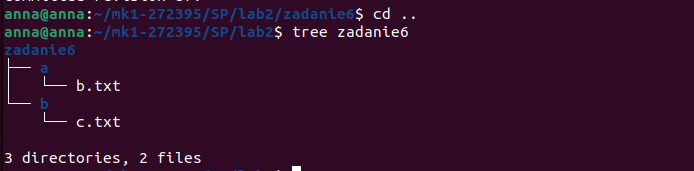
$ svn move b/a.txt b/c.txt

$ svn commit -m "Dopisanie wiersza do a.txt i zmiana nazwy na c.txt"



# Struktura katalogów po kroku 3:

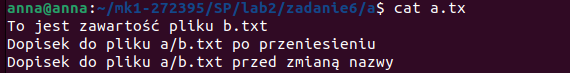
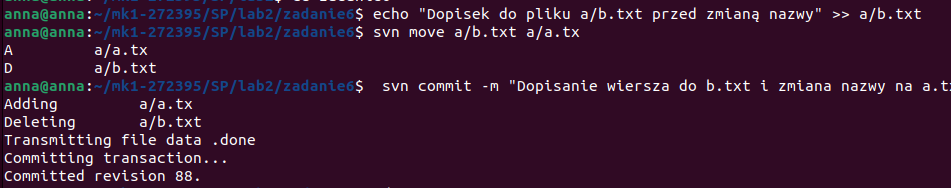
$ tree zadanie6



* **Krok 4: Dopisanie wiersza do b.txt i zmiana nazwy na a.txt**

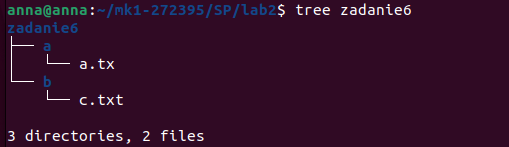
echo "Dopisek do pliku a/b.txt przed zmianą nazwy" >> a/b.txt

$ svn move a/b.txt a/a.txt

$ svn commit -m "Dopisanie wiersza do b.txt i zmiana nazwy na a.txt" 

# Struktura katalogów po kroku 4:

$tree zadanie6

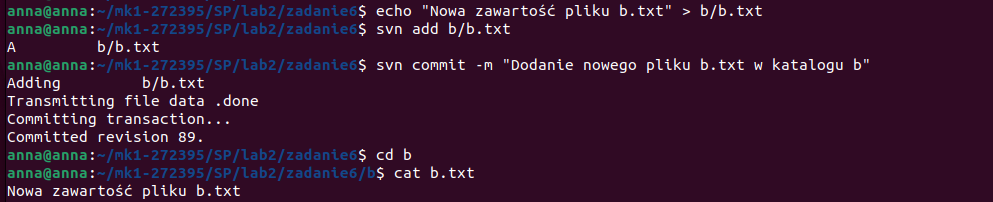


* **Krok 5: Dodanie nowego pliku b.txt w podkatalogu b**

$ echo "Nowa zawartość pliku b.txt" > b/b.txt

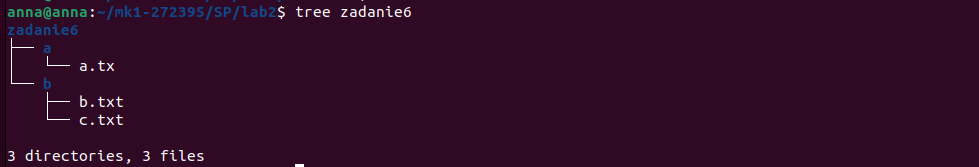
$ svn add b/b.txt

$ svn commit -m "Dodanie nowego pliku b.txt w katalogu b"



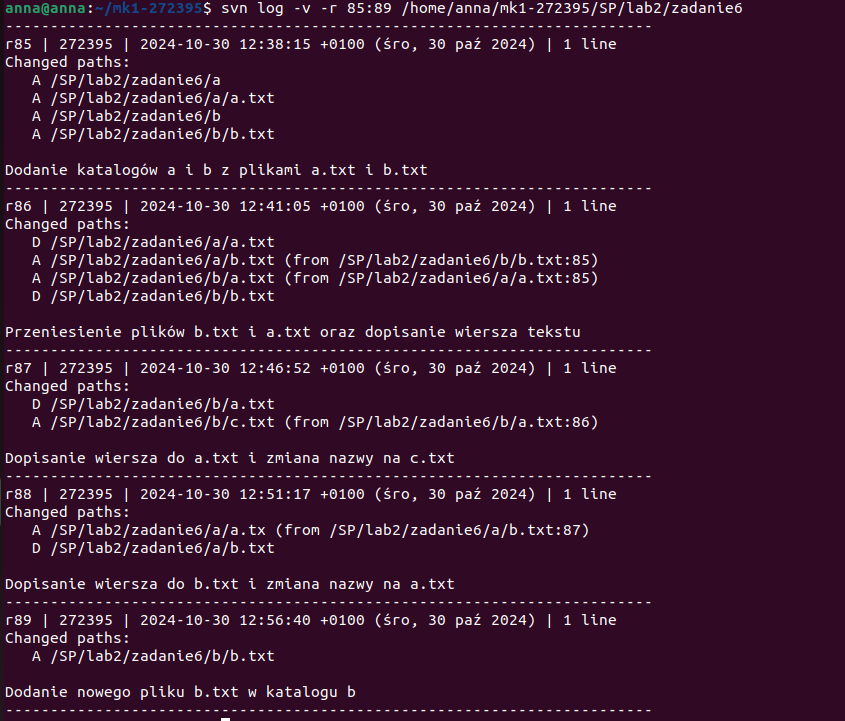
# Struktura katalogów po kroku 5:

$ tree zadanie6



# Ślad działań w repozytorium

$ svn log -v -r 85:89 /home/anna/mk1-272395/SP/lab2/zadanie6:



* **Demonstracja peg-revision i operative revision na przykładzie pliku `b/c.txt`**

* **1. Wyświetlenie zawartości pliku w konkretnej peg-revision (rewizja, w której plik istnieje)**

$ svn cat -r 86 b/a.txt@86

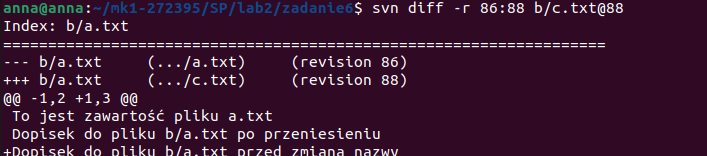
# Wyświetli zawartość pliku `a.txt` w rewizji 86, pokazując jego zawartość, mimo że nazwa pliku została zmieniona.



* **2. Wyświetlenie różnic między rewizjami z użyciem peg-revision i operative revision**

$ svn diff -r 86:88 b/c.txt@88

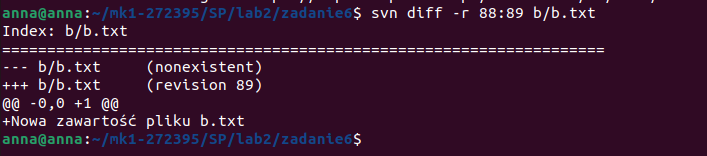
# Komenda ta analizuje zmiany między rewizją 86 a 88 w pliku `c.txt`, gdzie `@88` określa nazwę pliku w rewizji 88.



* **3. Sprawdzenie stanu pliku `b/b.txt` po jego dodaniu i pokazanie różnic między kolejnymi rewizjami.**

$ svn diff -r 88:89 b/b.txt

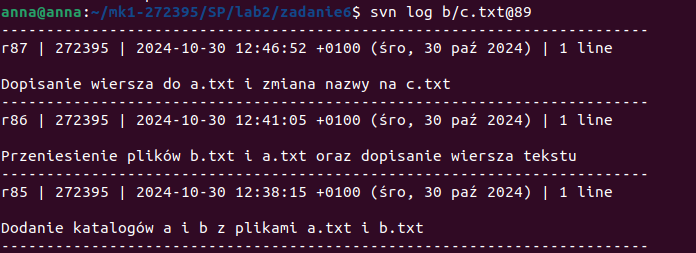
# Wyświetla różnicę między rewizją 88 a 89, gdzie plik `b.txt` został dodany w katalogu `b`.



* **4. Wyświetlenie historii pliku po jego przeniesieniu, z użyciem peg-revision**

$ svn log b/c.txt@89

# Historia pokazuje pełne zmiany i modyfikacje w `a.txt` aż do momentu zmiany nazwy na `c.txt` w rewizji 89.



* **Wyjaśnienie peg-revision i operative revision**

W SVN pojęcia \*pre-revision\* i \*operative revision\* odnoszą się do wersji, które są uwzględniane przy operacjach, takich jak `svn diff`. Oto czym się różnią:

1. Pre-revision:

- To rewizja, którą traktuje się jako punkt odniesienia lub „poprzednią” wersję przed wprowadzeniem zmian.

- Wykorzystuje się ją w celu określenia stanu elementu przed zastosowaniem modyfikacji.

- Na przykład, w poleceniu `svn diff -r 86:88`, pre-revision to rewizja 86, która przedstawia stan pliku przed wprowadzeniem zmian.

2. Operative revision:

- To rewizja, do której stosowane są modyfikacje lub z którą porównywany jest stan elementu. Jest to „docelowa” rewizja, w której znajdują się najnowsze zmiany w zakresie.

- Operative revision jest rewizją, która kończy porównanie lub operację.

- W poleceniu `svn diff -r 86:88`, operative revision to rewizja 88, która przedstawia stan pliku po wprowadzeniu zmian.