Konwencje

Skrypt, który właśnie czytasz zawiera komendy oraz lokalizacje/ścieżki oznaczone `monotypicznym` krojem pisma.

Pamiętaj aby podczas pracy zachować pisownię liter odpowiedniej wielkości!

Gromadź wyniki ćwiczeń i zadań.

Śmiało eksperymentuj z przedstawionymi ćwiczeniami i zadaniami.

W przypadku problemów zgłaszaj potrzebę wsparcia, tak aby wszystko było zrozumiałe.

Rozwiązywanie problemów będzie źródłem satysfakcji.

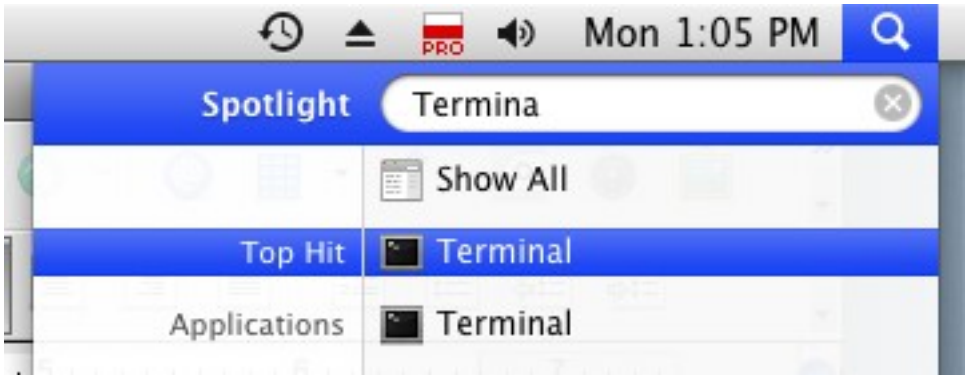
Wprowadzenie do konsoli OSX

Historia

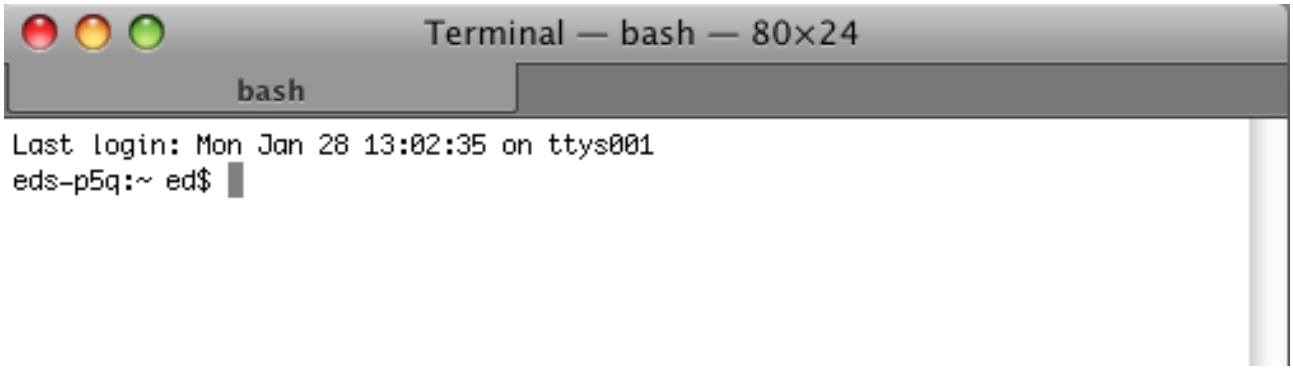
Podstawowym narzędziem komunikacji z systemami operacyjnymi UNIX lub systemami UNIX/-pochodnymi (np. Android, FreeBSD, iOS, LINUX, OSX) jest powłoka (tzw. shell) i jej konsola. W zależności od systemu operacyjnego, powłoki i ich konsole mogą się różnić, choć podstawowa idea i filozofia działania pozostaje ta sama: w trybie tekstowym, mając do wyboru pewien zakres komend, możemy formułować nasze potrzeby. O ile GUI (**G**rafical **U**ser **I**nterface), czyli interface graficzny jest zdecydowanie bardziej popularny, to znajomość komend pozwala na bardziej swobodną pracę i przypomina dialog z maszyną. Co więcej, tryb tekstowy przez niższe wymogi zdecydowanie lepiej sprawdza się w pracy zdalnej.

Ćwiczenie 1

W pasku Spotlight, w prawym górnym rogu ekranu, wpisz 'Terminal' i wybierz pierwszą z pozycji.



W ten sposób zostanie uruchomiona konsola powłoki.



W pierwszym wierszu znajduje się informacja o ostatnim zalogowaniu, w drugim wierszu jest podana nazwa użytkownika, nazwa stacji roboczej (czyli komputera na którym aktualnie pracujesz; porównaj z System Prefereces/Ustawienia Systemu > Sharing/Udostępnianie > Computer name/Nazwa komputera) oraz aktualna lokalizacja/ścieżka. Być może ten zapis jest na początku dość enigmatyczny, jednak w czasie łatwo do nieco przywyknąć.

Ćwiczenie 2

Aby uniknąć dezorientacji w nowym otoczeniu chciałbym wiedzieć **kim jestem** i **gdzie się znajduję**. W tym celu wpisz komendę `whoami` i wykonaj ją zatwierdzając klawiszem Return

A oto i wynik jaki otrzymałem:

```
eds-p5q:~ ed$ whoami
ed
```

Na pytanie gdzie jestem, mogę uzyskać odpowiedź poprzez komendę `pwd` (ang. print working directory):

```
eds-p5q:~ ed$ pwd
/Users/ed
```

Aby wyjaśnić ten drugi wynik warto dodać, że główny katalog w systemach UNIX/-pochodnych oznaczany jest symbolem `/`(tzw. `slash`, czyli `ciach`), kolejne lokalizacje odseparowane są również tym znakiem. Dlatego wynik `/Users/ed` można przeczytać: 'jestem w katalogu domowym użytkownika `ed` (katalog nazwy aktualnego użytkownika, np. Student), który to katalog znajduje się w katalogu `Users` (czyli katalogu wszystkich użytkowników). Z kolei katalog `Users` znajduje się w głównym katalogu.

Zauważ, że w systemach Microsoft stosuje się odwrotny znak do powyższych oznaczeń, czyli `\ backslash`, `w tył ciach`.

Ćwiczenie 3

Do zapoznania się z zawartością określonego katalogu służy polecenie `ls` (ang. list directory contents). W moim wypadku wynik tej komendy wygląda następująco:

```
eds-p5q:~ ed$ ls
Applications        Library                 Public
Desktop                 Movies              Send Registration
Documents           Music               Sites
Downloads           Pictures
```

Choć nazwy katalogów są 'samo-tłumaczące' wynik pozostaje sporym 'rozgardiaszem'. Chciałbym w przystępniejszej formie dowiedzieć się kilku informacji na temat moich katalogów. Dlatego też zastosuję polecenie `ls -l`.

A oto rezultat:

```
eds-p5q:~ ed$ ls -l
total 8
drwxr-xr-x   2 ed    staff    68 Sep  1  2011 Applications
drwx------+ 13 ed    staff   442 Jan 28 13:11 Desktop
drwx------+  9 ed    staff   306 Jan 21 15:59 Documents
drwx------+  4 ed    staff   136 Apr 15  2010 Downloads
drwx------+ 35 ed    staff  1190 Jan 21 18:21 Library
drwx------+  4 ed    staff   136 Oct 14  2010 Movies
drwx------+  4 ed    staff   136 Aug 13  2011 Music
drwx------+  4 ed    staff   136 Apr 15  2010 Pictures
drwxr-xr-x+  6 ed    staff   204 Aug 22  2011 Public
lrwxr-xr-x   1 root  staff    52 Apr 15  2010 Send Registration ->
/Users/ed/Library/Assistants/Send Registration.setup
drwxr-xr-x+  6 ed    staff   204 Aug 22  2011 Sites
```

Tym razem dane zostały zebrane w sposób tabelaryczny (szczegółowe wytłumaczenie powyższego listingu znajdziesz w podrozdziale Uprawnienia).

Rola konsoli

Choć powłoka wydaje się dość prymitywnym środowiskiem pracy, większość operacji związanych z pracą i administracją systemu można wykonać właśnie z jej poziomu. Co więcej, za pomocą powłoki można logować się zdalnie na inne maszyny, a otrzymując prawa administracyjne możemy zrobić praktycznie wszystko! Oczywiście potrzebna jest pewna wiedza na temat funkcjonowania systemów UNIX/-pochodnych, a także znajomość komend. Dzięki dostępnym funkcjom można łączyć polecenia, tak aby otrzymywać odpowiedzi na dość złożone zapytania, np.:
„znajdź pliki, które zostały zmodyfikowane w ciągu ostatnich 8 dni, mające określony rozmiar (np. mniejsze, większe) i utworzone przez użytkownika Student” - patrz podrozdział Łączenie komend.

Extra: Skąd wiem jakie komendy są dostępne w moim systemie? W celu odpowiedzi na to pytanie z poziomu powłoki należy dwukrotnie kliknąć klawisz Tab:

```
Display all 1673 possibilities? (y or n)
```

Po zatwierdzeniu (`y`) zostanie przedstawiony wykaz dostępnych poleceń systemowych. Aby kontynuować wyświetlanie kolejnych pozycji wystarczy zatwierdzić klawiszem Return. Aby zatrzymać wyświetlanie wystarczy wcisnąć klawisz q (ang. quit). Liczba dostępnych komend może się różnić w zależności od systemu.

'Moc' klawisza Tab.
Klawisz Tab pozwala nie tylko wyświetlić listę dostępnych komend ale też 'podpowiada' lub finalizuje niedokończone komendy, czy też nazwy katalogów/plików.

Ćwiczenie 4

1. Zależy mi na informacji o dostępnych kartach sieciowych, ew. przydzielonym im adresach. Komenda która będzie tu pomocna to `ifconfig`.
2. Wpisz literę `i`, a następnie wciśnij klawisz Tab.
3. No cóż – mało satysfakcjonująco. Jak widać komend rozpoczynających się na tę literę jest całkiem sporo. Przerwij wyświetlanie na podstawie zdobytej już wiedzy.
4. W linii komend pozostaje wciąż wprowadzona litera `i`, dopisz zatem `f`, a następnie wciśnij klawisz Tab:

```
eds-p5q:~ ed$ if
if        ifconfig
eds-p5q:~ ed$ if
```

5. Tym razem jest zdecydowanie lepiej – są już tylko dwie możliwości.
6. Jeżeli więc dopiszemy `c`, tak aby uzyskać `ifc`, i wciśniemy klawisz Tab, reszta zostanie dodana automatycznie (nie ma już ambiwalencji).
7. Zatwierdź klawiszem Return prowadzoną komendę `ifconfig`.

Ćwiczenie 5

1. Zależy mi na sprawdzeniu zawartości katalogu `Library`. Tym razem wykorzystam klawisz Tab, aby dokończył wpisywanie nazwy dla tego katalogu (ten przykład działa również dla plików, linków, itd.).
2. Wpisz `ls -l L`, a następnie wciśnij klawisz Tab. O ile nie występują w tej lokalizacji inne katalogi/pliki, których nazwy zaczynają się na literę L, nazwa Library zostanie automatycznie wprowadzona.

Wróćmy do przykładów z Ćwiczenia 3. Skąd wiadomo, że akurat `ls -l` (czyli komenda `ls` z parametrem `l`) jest w tym wypadku odpowiednia? Dla każdej z komend powłoki dostępny jest podręcznik `man` (ang. manual). Aby uzyskać więcej informacji dla komendy `ls` wystarczy posłużyć się następującą składnią:

```
man ls
```

Za pomocą klawiszy kursora (góra/dół) można nawigować po stronach podręcznika. Aby odnaleźć określoną frazę wystarczy wcisnąć klawisz / a następnie wprowadzić poszukiwaną frazę i zatwierdzić klawiszem Return.
Wśród dostępnych opcji podręcznika dla komendy `ls` znajduje się między innymi `h` (znana dla wielu innych funkcji jako ang. human readable). Mając tę wiedzę wypróbuj działanie komendy `ls` z kilkoma innymi parametrami (pamiętaj, że parametry możesz je łączyć).

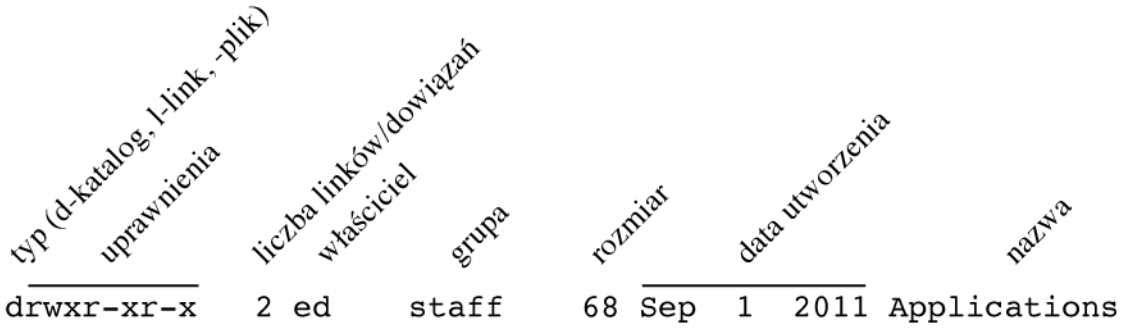
Zadanie 1

Na podstawie komendy podręcznika `man`, ew. pomocy (`ls -help`) sprawdź strukturę katalogów/plików w głównym katalogu.

Ciekawostka: Struktura systemu plików dla UNIX/-pochodnych systemów jest podobna. O ile ktokolwiek jest zainteresowany głębszym poznaniem własnego smartphone'a lub tablet'u (w szczególności urządzenia oparte na systemach Android, iOS) nabycie uprawnień administracyjnych (tzw. jailbrake, root) pozwoli również na odkrycie podobnego systemu plików, który dla zwykłych użytkowników jest zablokowany. Jednak uzyskanie takiego dostępu z reguły równoznaczne jest z utratą gwarancji na sprzęt.

Uprawnienia

Rezultat wykonanego w Ćwiczeniu 3 polecenia `ls -l` można wytłumaczyć na przykładzie katalogu `Applications`:



Ryc. 1. Diagram atrybutów katalogu uzyskany przez polecenie `ls -l`.

Pierwsza kolumna zawiera informacje dotyczące uprawnień dla poszczególnych katalogów/plików. Takie rozwiązanie daje możliwość odpowiedniego zabezpieczania systemu. Przyjrzyjmy się tej kolumnie, znajduje się w niej następujący zapis:

```
drwxr-xr-x
```

Pierwsza litera, czyli `d` oznacza, że mamy do czynienia z katalogiem (ang. directory). Dalsza część zapisu dotyczy trzech typów użytkowników i można ją przestawić jako:

```
drwxr-xr-x
właściciel grupa pozostali
```

Poza pierwszą pozycją `d` (typ, patrz Ryc1.) pozostałe 9 pozycji może zawierać następujące znaki, które nadają uprawnienia do:

`r` – odczytu (ang. read)
`w` – zapisu (ang. write)
`x` – wykonywania/uruchamiania (ang. execute)
`-` – brak uprawnień

Extra: W systemach UNIX/-pochodnych można utworzyć wiele kont dla użytkowników. Z kolei każdego z użytkowników można przypisać do poszczególnych grup. Grupy zawierają własne uprawnienia, np. grupa uprawniona do używania sprzętu (nośników, kart dźwiękowych/graficznych, drukarek itd.). Dlatego też precyzyjnie, poprzez przynależność do odpowiednich grup, nadawane są prawa poszczególnym użytkownikom.

Ćwiczenie 6

Do zmiany atrybutów uprawnień dla katalogów/plików służy polecenie `chmod`. Aby sprawdzić jaka działa to polecenie potrzebujmy eksperymentalny plik.

1. Utwórz plik o dowolnej nazwie poprzez polecenie: `>dowolna_nazwa_pliku`.
2. Sprawdź czy plik został faktycznie utworzony za pomocą poznanej już komendy.
3. Za pomocą polecenia `chmod` możesz nadawać i odbierać uprawnienia. Na podstawie poniższych przykładów, sprawdź za każdym razem, czy coś się zmieniło w uprawnieniach dla pliku:

```
chmod u+x dowolna_nazwa_pliku
chmod g+rw dowolna_nazwa_pliku
chmod o-w dowolna_nazwa_pliku
chmod a+rwx dowolna_nazwa_pliku
```

Przedstawione przykłady operują symbolami:
u – user, czyli właściciel pliku
g – group (grupa)
o – other (pozostali)
+/- – to przyznawanie lub odmawianie uprawnień

Dla polecenia `chmod` istnieje również skrócona i przez to bardziej popularna notacja, którą można wyrazić za pomocą liczb:

```
0 ---
1 --x
2 -w-
3 -wx
4 r--
5 r-x
6 rw-
7 rwx
```

Stosując tę notację polecenie `chmod` może przyjąć np. postać:

```
chmod 755 dowolna_nazwa_pliku
```

Zadanie 2
Utwórz 4 pliki (o nazwach: `plik1.txt, plik2.txt, plik3.txt, plik4.txt`) i za pomocą notacji numerycznej polecenia `chmod` ustal uprawnienia jak w Ćwiczeniu 6. Nie usuwaj tych plików!

Przekierowanie wejścia/wyjścia

Zastanówmy się nad poleceniem użytym w Ćwiczeniu 6, które spowodowało powstanie pliku:

```
>dowolna_nazwa_pliku
```

Otóż użyty znak `>` to tzw. przekierowanie dla wyjścia (ang. output redirection). Ponieważ nie mieliśmy żadnych danych (nie było wejścia, ang. input) powstał pusty plik. Jak sprawdzisz, czy otrzymany plik faktycznie jest pusty?

Ćwiczenie 7
Wypełnijmy utworzone z Zadaniu 2 pliki txt. Do tego calu posłużymy się komendą `echo`, która 'drukuje' podaną w niej treść. Na początek:

```
eds-p5q:~ ed$ echo 'Czy to naprawdę działa?'
Czy to naprawdę działa?
```

No cóż, efekt może nie poraża, ale komenda echo i przekierowania wyjścia mają już pewien sens:

```
echo 'Czy to naprawdę działa?' > plik1.txt
```

Jak widać ten plik różni się rozmiarem od pozostałych:

```
-rw-r--r--  1 ed  staff  26 Jan 30 19:34 plik1.txt
-rw-r--r--  1 ed  staff   0 Jan 30 19:34 plik2.txt
-rw-r--r--  1 ed  staff   0 Jan 30 19:34 plik3.txt
-rw-r--r--  1 ed  staff   0 Jan 30 19:34 plik4.txt
```

W jaki sposób jednak sprawdzić treść? Do tego celu można użyć polecenia `cat` (ang. concatenate and print files):

```
eds-p5q:~ ed$ cat plik1.txt
Czy to naprawdę działa?
```

Załóżmy, że chcielibyśmy dodać treść. W tym celu kolejny raz możemy użyć zmodyfikowanej komendy przekierowania wyjścia. Jednak zamiast mozolnie wpisywać tę komendę, skorzystaj z pamięci konsoli i za pomocą górnego klawisza kursora znajdź komendę:

```
echo 'Czy to naprawdę działa?' > plik1.txt
```

a następnie zmodyfikuj ją jak niżej:
```
echo 'Czy to naprawdę działa?' >> plik1.txt
```

Sprawdź za pomocą polecenia `cat` czy doszło do wprowadzenia zmian.

Teraz wprowadź komendę:
```
echo 'A czy to działa?' > plik1.txt
```
a następnie sprawdź zawartość pliku.

Ja widać w efekcie stosowania przekierowania `>` nadpisywany jest cały plik! Podczas, gdy użycie `>>` powoduje dopisanie treści do pliku. To zasadnicza różnica – pamiętaj o tym, o ile nie chcesz stracić efektów swojej pracy!

Zadanie 3
Znając zasadę przekierowania wyjścia, za pomocą znanych już komend utwórz plik (np. o nazwie `zawartosc_katalogu`), który będzie zawierał wykaz (w formie tabelarycznej) zawartości katalogu, w którym aktualnie pracujesz. Następnie do tego pliku dodaj lokalizację/ścieżkę. Zachowaj ten plik!
W sumie w twoim katalogu domowym powinny znajdować się poniższe pliki:
```
|__plik1.txt
|__plik2.txt
|__plik3.txt
|__plik4.txt
|__zawartosc_katalogu
```

Ścieżki absolutne i relatywne

Ścieżka to zapis, która za pomocą składni opisuje konkretną lokalizację. Nie sposób więc swobodnie poruszać się w systemie katalogów/plików nie znając ścieżek.

Dla przykładu, wykonanie polecenia `pwd` może dać dyskutowany już wynik:

```
/Users/ed
```

Tak zapisany wynik to tzw. **ścieżka absolutna**, zawiera bowiem wszystkie informacje, od katalogu głównego włącznie, po kolejne katalogi, ew. pliki. Mogę dowiedzieć się więcej o zawartości tej lokalizacji podając właśnie jej absolutną ścieżkę:
```
ls -l /Users/ed
```

Jednak, jeżeli obecnie przebywam w tej lokalizacji, czyli `/Users/ed` (o czym świadczy wynik polecenia `pwd`) wystarczy, że wpiszę:
```
ls
```
lub nieco precyzyjniej
```
ls ./
```
Komentarz: W praktyce nie ma różnicy między zapisem `ls` oraz `ls ./`. Oba wyrażenia znaczą to samo – 'wykonaj `ls` dla katalogu, w którym obecnie znajduję się'. Jednak zapis `ls ./` jest bardziej precyzyjny, ponieważ `./` jest uściśleniem ścieżki 'aktualny katalog'.

Po wykonaniu poleceń `ls` lub `ls ./` wynik będzie ten sam, co w przypadku zastosowania ścieżki absolutnej (`ls /Users/ed/`), ale notacja jest zdecydowanie krótsza, gdyż została zastosowana **ścieżki relatywna**, **względna**, czyli taka której punktem odniesienia jest **aktualna lokalizacja**.

Gdybym chciał uzyskać informację na temat zawartości katalogu `Users`, na zasadzie analogii mogę wykonać komendę jak niżej:
```
ls -l /Users/
```

Ten zapis wyrażony jest za pomocą ścieżki absolutnej. Jak to jednak zrobić za pomocą ścieżki relatywnej, zakładając, że wciąż przebywam w katalogu podrzędnym do `Users`, np. `ed` czy `Student`?

Powinienem 'wspiąć się' jeden poziom wyżej, czyli:
`ls ../`

O ile mamy odpowiednie uprawnienia można 'podróżować' kilka poziomów do góry, np.:

`ls ../../`

Extra: Dla niektórych dystrybucji UNIX/-pochodnych, w tym m. in. dla OSX czy Ubuntu, możliwe jest odniesienie się do własnego katalogu domowego za pomocą skrótu `~/`, czyli zamiast:

`ls /Users/ed/`

wystarczy wykonać:

`ls ~/`

W ten sposób można za pomocą krótkiej notacji odnieść się do własnego katalogu domowego (np. skopiować do niego pliki, itd.). Sprawdź, czy system na którym pracujesz oferuje możliwość użycia tej krótkiej notacji: `~/`.

Ćwiczenie 8
Zakładając, że jesteś w katalogu domowym użytkownika (np. `ed`, `Student`, itd.) oraz że ten katalog zawiera katalog (np. `Library`), zaproponuj komendę, która sprawdzi zawartość tego katalogu używając do tego celu ścieżki absolutnej, ścieżki relatywnej oraz skróconej notacji `~/`.

Podstawowe komendy do operacji na katalogach/plikach

Zanim na dobre zaczniemy zmieniać obecną strukturę katalogów i plików zatroszczmy się o wolną od poprzednich komend przestrzeń. W celu 'wyczyszczenia' terminala użyj polecenia `clear`, a wszystkie wprowadzone wcześniej komendy, ew. ich skutek zniknie z ekranu (*de facto* można wciąż je podglądnąć korzystając ze scroll'a).

To co przeważnie potrzebujemy do organizowania własnej pracy, to tworzenie, usuwanie, kopiowanie, ew. przenoszenie katalogów i plików.

Tworzenie katalogów

Już wiesz jak utworzyć plik. Aby utworzyć katalog należy wykonać `mkdir` (ang. make directory):
`mkdir /lokalizacja/dowolna_nazwa_katalogu`
O ile mamy prawa zapisu w lokalizacji (patrz Uprawnienia) zostanie utworzony katalog.

Przykład 1:

Na podstawie poniższego listingu komend zastanów się dlaczego nie udało mi się utworzyć katalogu test:

```
eds-p5q:~ ed$ >test
eds-p5q:~ ed$ mkdir ./test
mkdir: test: File exists
eds-p5q:~ ed$
```

Usuwanie katalogów/plików

Do usuwania służy polecenie `rm` (ang. remove):

```
rm nazwa_dowolnego_pliku
```

Utwórz dowolny plik, a następnie usuń go.

Czas na usuwanie katalogów. W pierwszej kolejności przeanalizuj podany niżej przykład

Przykład 2:

```
eds-p5q:~ ed$ mkdir ./test
eds-p5q:~ ed$ rm ./test
rm: test: is a directory
eds-p5q:~ ed$
```

Zauważ, że nie udało mi się usunąć katalogu `test`. Aby to zrobić powinienem zmodyfikować komendę `rm`, tak aby określić, że mam na myśli usunięcie całego katalogu, a więc i zawartej w nim treści (podkatalogów, plików). Poprawna komenda powinna wyglądać następująco:

```
rm -r test
```

W tym wypadku została dodana opcja/parametr `r` (ang. recursive, czyli rekursywnie/'równocześnie'). Powyższy przypadek pokazuje też składnię typową dla wielu innych komend operujących na katalogach/plikach, która wg. podręcznika (`man rm`) została opisana i przedstawiona następująco:

```
SYNOPSIS
     rm [-dfiPRrvW] file ...
```

Kopiowanie i przenoszenie katalogów/plików

`cp` (ang. copy) jest komendą odpowiedzialną za kopiowanie. Podobnie jak w przypadku komendy `rm`, rozróżniamy kopiowanie plików oraz katalogów ('opcja rekursywna' `-r`). Ogólna składnia dla polecenia cp to:

```
cp /lokalizacja/źródłowa /lokalizacja/docelowa
```

`mv` (ang. move) przenosi katalogi/pliki. W przeciwieństwie do `cp`, dla `mv` nie jest wymagane używanie opcji rekursywnej, a składnia wygląda jak niżej:

```
mv /lokalizacja/źródłowa /lokalizacja/docelowa
```

Jeżeli obie lokalizacje (źródłowa i docelowa) są takie same, to przeniesienie katalogu/pliku skończy się zmianą jego nazwy.

Ćwiczenie 9

Utwórz kilka katalogów oraz plików i wypróbuj dla nich działanie komend `cp` oraz `mv`. O ile nie masz pomysłu na ich zastosowanie skorzystaj z podręcznika `man`.

Zadanie 4

1. W katalogu domowym Student utwórz własny katalog. Nazwa nie powinna zawierać znaków diakrytycznych, powinna być podzielona na 3 człony, zawierać małe litery, podkreślenia zamiast spacji – jak w poniższym przykładzie:
   `latex_krzysztof_moszczynski`

2. W katalogu utworzonym w pkt. 1 utwórz katalogi:
   `cwiczenia`, `pliki`, `zadania`
   `zadania/03`

   Podpowiedź: Do poruszania się (zmiany lokalizacji) służy komenda `cd` (ang. change directory). Aby wejść do określonej lokalizacji należy wykonać:
   `cd nazwa_dowolnego_istniejącego_katalogu`
   Jeżeli chciałbym wejść poziom wyżej należy wykonać komendę:
   `cd ../`

3. Przenieś utworzone wcześniej (Zadanie 2) pliki: `plik1.txt...plik4.txt`, (Zadanie) plik:
   Całość powinna mieć strukturę jak przykładzie poniżej:
   Przykład 3:

```
latex_krzysztof_moszczynski
|__cwiczenia
|__pliki
| |__plik1.txt
| |__plik2.txt
| |__plik3.txt
| |__plik4.txt
|__zadania
   |__03
     |__zawartosc_katalogu
```

- Ustaw uprawnienia dla katalogu `latex_imie_nazwisko` oraz zawartych w nim katalogów/plików tak, aby odczyt, zapis i wykonywanie były dostępne tylko dla użytkownika Student.

  Podpowiedź: Aby zaoszczędzić mozolnej pracy, da polecenia `chmod` użyj opcji rekursywnej.

4. Skopiuj swój plik do repozytorium Time Capsule, do własnego katalogu (o ile nie dysponujesz takim katalogiem skonsultuj się z prowadzącym).

   Podpowiedź: Repozytorium Time Capsule montowane jest w lokalizacji `/Volumes/data/`. Aby zorientować się w strukturze podczas wykonywania komendy `cp` posługuj się klawiszem Tab.

Wyrażenia regularne

Każdy z nas dla automatyzacji pracy, np. podczas nanoszenia korekt w edytorach tekstu, posługuje się komendą znajdź, ew. znajdź/zamień. Wyrażenia regularne (ang. regular expressions) zostały stworzone właśnie do tego celu i do dziś stanowią istotny element programowania. Choć zgłębianie całego zagadnienia wykracza poza ramy tego kursu, warto poznać choć kilka przykładów 'liter' tego 'alfabetu', tak aby ułatwić sobie pracę w wielu środowiskach (np. konsoli powłoki, edytorach sed/vim, a nawet w pakietach LibreOffice):

`.` - zaznacza jeden znak

`*` - zaznacza zero lub kilka wystąpień wcześniejszych znaków

`[]` - zaznacza zakres, np. [0-9] lub [tTxXtT]

`dowolne_znaki` – zaznaczy dokładnie takie wyrażenie, czyli: 'dowolne_znaki'

Ćwiczenie 9

Zakładam, że dysponujesz strukturą katalogów/plików podaną w Przykład 3. W innym razie zbuduj taką strukturę.

1. Przejdź do katalogu `pliki`.
2. Wypełnij każdy z plików inną treścią (podpowiedź: Ćwiczenie 7), tak aby plik nie były puste.
3. Wykonaj komendę `cat * > all.txt`
4. Sprawdź wielkość tego pliku oraz jego zawartość.

Komenda wykonana w pkt. 3 'zaznaczyła' wszystkie pliki, a następnie ich treść została przekierowana do jednego pliku wynikowego: `all.txt`. To druga funkcja komendy `cat`, czyli łączenie plików (ang. `contatenate`).

Chciałbym jeszcze raz otrzymać plik o takiej samej treści (a więc i rozmiarze) co `all.txt`, ale o nowej nazwie, np. `all_1.txt`. Nie mogę tego wykonać przez komendę:

```
cat * > all_1.txt
```

bo przecież zostaną uwzględnione wszystkie pliki, także `all.txt`. W istocie pliki, które zamierzam połączyć mają wspólny rdzeń nazwy, mianowicie: `plik` i jest to właśnie rozwiązanie całego problemu:

```
cat plik* > all_1.txt
```

Zadanie 5

Za pomocą powyższych wyrażeń regularnych przenieś wszystkie pliki o nazwie plik* jeden poziom wyżej, tak aby struktura wyglądała następująco:

```
latex_krzysztof_moszczynski
|__cwiczenia
|__plik1.txt
|__plik2.txt
|__plik3.txt
|__plik4.txt
|__pliki
| |__all.txt
| |__all_1.txt
|__zadania
   |__03
      |__zawartosc_katalogu
```

Zaproponuj dwa wyrażenia regularne, aby przenieść te pliki do pierwotnej lokalizacji, czyli do folderu `pliki`.

Łączenie komend

Komendy powłoki można ze sobą łączyć, ew. przekazywać ich wynik kolejnym komendom. Do tego celu posłużymy się symbolem `|` (ang. pipe). Przeanalizujmy znany już przykład polecenia `ls -l`:

```
…
drwxr-xr-x+  6 ed    staff   204B Aug 22  2011 Sites
drwxr-xr-x   3 ed    staff   102B Jan 31 15:53 bin
drwxr-xr-x   9 ed    staff   306B Feb  1 17:57 latex_krzysztof_moszczynski
-rw-r--r--   1 ed    staff   1.8K Feb  1 16:39 writer2latex.xml
…
```

Otóż nie jestem zainteresowany całym listingiem, ale tylko katalogami, które zawierają frazę 'latex', np. `latex_krzysztof_moszczynski`. W tym celu mogę 'przechwycić' wynik i 'odfiltrować' go za

pomocą polecenia `grep`:

```
eds-p5q:~ ed$ ls -l | grep latex
drwxr-xr-x   9 ed    staff   306 Feb  1 17:57 latex_krzysztof_moszczynski
-rw-r--r--   1 ed    staff  1869 Feb  1 16:39 writer2latex.xml
```

Jak powyżej widać, `grep` przechwycił wszystkie wyrażenia (tak, to także filozofia wyrażeń regularnych), które zawierają frazę 'latex'.

Kolejnym przykładem łączenia komend jest polecenie `find`. Jak wskazuje nazwa, za pomocą `find` możemy znaleźć katalog/plik. Dodatkowo poszukiwana lokalizacja może spełniać określone kategorie (więc na ten temat na stronach `man`). Zastanówmy się nad dyskutowanym już wcześniej zapytaniem: „znajdź pliki, które zostały zmodyfikowane w ciągu ostatnich 8 dni, mają określony rozmiar (np. mniejsze, większe) i zostały utworzone przez użytkownika Student”. Aby zawęzić wynik poszukiwań w poniższym przykładzie ograniczę się tylko do jednego katalogu, ew. do umieszczonych w nim innych katalogów/plików:

```
find ./latex_krzysztof_moszczynski -mtime -3 -user ed
./latex_krzysztof_moszczynski
./latex_krzysztof_moszczynski/latex_krzysztof_moszczynski.tar.gz
./latex_krzysztof_moszczynski/pliki
```

Powyższa komenda zawiera opcję `-mtime` (ang. modification time) i parametr `-3` (czyli mniej niż 3 dni, wg. `man` możesz zmodyfikować te wartości np. do sekund, minut, godzin, itd.). Dodatkowo poprzez opcję `-user` został określony użytkownik (czyli `ed`).

Ćwiczenie 10

Zmodyfikuj powyższą komendę tak aby była dostosowana do dyskutowanego wyżej zapytania i dotyczyła twojego katalogu `latex_*` (symbol '`*`' jest oczywiście użyty w kontekście wyrażenia regularnego). Pamiętaj, aby w poniższych przykładach stosować własne zapytanie.

Dokładna analiza wyniku otrzymanego przez użytą komendę `find` wskazuje na 2 katalogi i jeden plik archiwum, które zostały zmodyfikowane (w moim wypadku w ciągu ostatnich 3 dni). Jednak ten wynik nie jest szczegółowy – przecież wiem, że powyższe katalogi dodatkowo zawierają pliki i to właśnie tymi plikami jestem zainteresowany (z reguły szukamy zmodyfikowanych plików, a nie katalogów). Zaglądnijmy więc nieco głębiej:

```
find ./latex_krzysztof_moszczynski -mtime -3 -user ed -type f
./latex_krzysztof_moszczynski/latex_krzysztof_moszczynski.tar.gz
./latex_krzysztof_moszczynski/pliki/all.txt
./latex_krzysztof_moszczynski/pliki/all_1.txt
./latex_krzysztof_moszczynski/pliki/plik1.txt
./latex_krzysztof_moszczynski/pliki/plik2.txt
```

Tym razem jest zdecydowanie lepiej, bo poszukiwałem tylko plików: `-type f` (ang. file). Dla plików, które spełniają kryteria wyszukiwania mogę wykonać jakąś komendę, np. skopiować je, usunąć, itd. W tym przypadku sprawdzę dokładnie ich rozmiar:

```
find ./latex_krzysztof_moszczynski -mtime -3 -user ed -type f -exec ls -lh {} \;
-rw-r--r--  1 ed  staff   427B Feb  8 20:06
./latex_krzysztof_moszczynski/latex_krzysztof_moszczynski.tar.gz
-rw-r--r--  1 ed  staff    64B Feb  9 12:58 ./latex_krzysztof_moszczynski/pliki/all.txt
-rw-r--r--  1 ed  staff    64B Feb  9 12:58 ./latex_krzysztof_moszczynski/pliki/all_1.txt
-rw-r--r--  1 ed  staff    52B Feb  9 12:59 ./latex_krzysztof_moszczynski/pliki/plik1.txt
-rw-r--r--  1 ed  staff     3B Feb  9 12:59 ./latex_krzysztof_moszczynski/pliki/plik2.txt
```

Mimo dość długiej komendy jak dotąd jeszcze nie łączyłem jej (nie przekazywałem wyniku jej działania) z inną komendą poprzez tzw. pile, czyli znak '|'. Załóżmy, że chciałbym otrzymać wykaz tych plików, tak aby był posortowany rosnąco wg ich rozmiaru:

```
find ./latex_krzysztof_moszczynski -mtime -3 -user ed -type f -exec ls -lh {} \; | sort
-rw-r--r--  1 ed  staff     3B Feb  9 12:59 ./latex_krzysztof_moszczynski/pliki/plik2.txt
-rw-r--r--  1 ed  staff    52B Feb  9 12:59 ./latex_krzysztof_moszczynski/pliki/plik1.txt
-rw-r--r--  1 ed  staff    64B Feb  9 12:58 ./latex_krzysztof_moszczynski/pliki/all.txt
-rw-r--r--  1 ed  staff    64B Feb  9 12:58 ./latex_krzysztof_moszczynski/pliki/all_1.txt
```

Zamykając zagadnienie komendy `find` warto dodać, że kryteriów poszukiwania może być zdecydowanie więcej (np. `-size`). W poszukiwaniach często nas interesuje zawartość plików i tu z pomocą przychodzi już znany grep:

```
find ./latex_krzysztof_moszczynski -mtime -3 -user ed -type f -exec grep -H -n 'Czy'
{} \;
./latex_krzysztof_moszczynski/pliki/all.txt:1:Czy to naprawdę działa?
./latex_krzysztof_moszczynski/pliki/all.txt:2:Czy to naprawdę działa?
./latex_krzysztof_moszczynski/pliki/all_1.txt:1:Czy to naprawdę działa?
./latex_krzysztof_moszczynski/pliki/all_1.txt:2:Czy to naprawdę działa?
./latex_krzysztof_moszczynski/pliki/plik1.txt:1:Czy to naprawdę działa?
./latex_krzysztof_moszczynski/pliki/plik1.txt:2:Czy to naprawdę działa?
```

Jak widać powyższy wynik zawiera informacje dot. pliku, nr wiersza, w którym znajduje się poszukiwana fraza 'Czy'. Oczywiście poszukiwania mogą być zdecydowanie bardziej zaawansowane, a Apple wyposaża dziś OSX w bardzo wyrafinowane narzędzie GUI do poszukiwania – sprawdź Finder > File/Plik > Find/Znajdź. Jednak pracując zdalnie, na różnych dystrybucjach UNIX/-pochodnych, takie narzędzie w większości nie jest dostępne i tu z pomocą przychodzi znajomość linii komend.

Tworzenie archiwów

Innym praktycznym przykładem łączenia komend może być tworzenie archiwów (tzw. kompresowanie/pakowanie katalogów/plików). Do tworzenia archiwum służy polecenie `tar`

o składni:

```
tar -cfp /ścieżka/nazwa_archiwum.tar /ścieżka/nazwa_katalogu_źródłowego
```

Co wg. podręcznika man znaczy `c` (create), `f` (file), `p` (preserve permissions), a więc utwórz plik archiwum `tar` zachowując uprawnienia (katalogów/plików)

Konkretniej tę składnię można przedstawić:

```
tar -cfp ./latex_krzysztof_moszczynski.tar ./latex_krzysztof_moszczynski/
```

W efekcie powstał plik archiwum tar:

```
-rw-r--r-- 1 ed staff 10K Feb 7 18:38 latex_krzysztof_moszczynski.tar
```

Ten plik możemy dodatkowo skompresować za pomocą gzip:

```
gzip ./latex_krzysztof_moszczynski.tar
```

A oto i wynik:

```
-rw-r--r-- 1 ed staff 427B Feb 7 18:38 latex_krzysztof_moszczynski.tar.gz
```

Jak widać udało się osiągnąć całkiem przyzwoity stopień kompresji, zamiast 10 KB (a więc 10 000B) otrzymałem 427B, a więc ponad 20 razy mniej!

Zadanie 6

Przedstawiona wyżej metoda tworzenia archiwów jest dość uciążliwa. Zamiast tworzyć archiwum tar, a następnie je kompresować możemy połączyć działanie obu komend, tzn. `tar` i `gzip`.

Zaproponuj rozwiązanie i zapisz jako plik w poniższej lokalizacji:
`latex_*/zadania/06/komenda_tar_gzip`

Ćwiczenie 11
Tworzenie archiwów jest na tyle powszechnym działaniem, że polecenie `tar` ma wbudowaną opcję kompresji, całość komendy wygląda:
`tar -czvpf ścieżka/nazwa_archiwum.tgz ścieżka/nazwa_katalogu_źródłowego`

Zwróć uwagę, że tym razem jest tylko jedno rozszerzenie (`.tgz`, zamiast `.tar.gz`). W wypadku powyższej komendy występują dodatkowe parametry: `z` oraz `v`. Sprawdź w `man`, co one oznaczają.

W oparciu o zdobyta wcześniej wiedzę użyj komendy, aby utworzyć archiwum w poniższej lokalizacji:
`latex_*/cwiczenia/11/latex_*.tgz.`

W toku wszystkich ćwiczeń i zadań twój katalog latex_* powinien mieć następującą strukturę:
```
|__
| |__cwiczenia
| | |__11
| | | |__latex_krzysztof_moszczynski.tgz
| |__pliki
| | |__all.txt
| | |__all_1.txt
| | |__plik1.txt
| | |__plik2.txt
| | |__plik3.txt
| | |__plik4.txt
| |__zadania
| | |__03
| | | |__zawartosc_katalogu
| | |__06
| | | |__komenda_tar_gzip
```

Dekompresja archiwów

Wielu użytkowników platform Apple do (de)kompresji używa zewnętrznych programów, np. Zipeg. Warto jednak podkreślić, że do rozpakowywania archiwów zip jest dostępne z linii komend polecenie `unzip`. Możesz zatem z poziomu powłoki używać tej komendy:
`unzip nazwa_archiwum.zip`

Pojawia się jednak pytanie w jaki sposób dekompresować archiwa, które tworzyliśmy wcześniej. Zarówno dla `.tar.gz` jak i `.tgz` używamy:
`tar -xzf ścieżka/nazwa_archiwum`
Można oczywiście użyć omawianych parametrów `v` oraz `p`. Zastanów się dlaczego te parametry nie zostały tu użyte.

Ćwiczenie 12
Utwórz dowolne archiwa `.zip`, `.tar.gz`, `.tgz` i poeksperymentuj z ich dekompresją.

Praca własna
Z oficjalnych zasobów LaTeX (http://www.latex-project.org/, ftp://ctan.tug.org/tex-archive/info/lshort) pobierz podręcznik 'Nie za krótkie wprowadzenie do systemu LATEX 2ε', tłum. Przechlewski T., Kubiak R., Gołdasz J. Warto podkreślić, że tych zasobach dostępny jest również oryginał 'The Not So Short Introduction to LATEX 2ε' autorstwa Oetiker T., Partl H., Hyna I., Schleg E. W zależności od własnych preferencji zapoznaj się z treścią pierwszego rozdziału polskiego tłumaczenia lub wybraną wersją językową.

Sesja LaTeX

Znajomość komend powłoki pozwala na nieskrępowaną pracę w środowisku, które reguły nie konsumuje zasobów więc jest bardzo efektywne, ponadto zestandaryzowane i uniwersalne w ramach systemów UNIX/-pochodnych.

Edytor vim

Poznaliśmy już sposób na 'podgląd' treści pliku poprzez polecenie `cat`. A co z potrzebami edycji? Do tego celu UNIX/-pochodne systemy wyposażone są w kilka znakomitych edytorów, m. in. `nano`, `emacs` oraz `vi`. Rodowód tych narzędzi czasami sięga lat 70-tych, a wyśmienita funkcjonalność czyni jest powszechnymi (szczególnie w środowiskach administratorów) do dnia dzisiejszego. Dla potrzeb tego kursu skorzystamy z 'poprawionej' wersji `vi`, czyli `vim` (Vi IMproved).

Praca własna
Zapoznaj się z pracą w środowisku `vim` poprzez samouczek dostępny z linii komend: `vimtutor`.

Zadanie 7
Utwórz za pomocą `vim` plik o nazwie `plik_vim`. W trybie edycji `vim` wprowadź dowolną treści i zapisz zmiany do lokalizacji:
`latex_*/zadania/07/plik_vim`

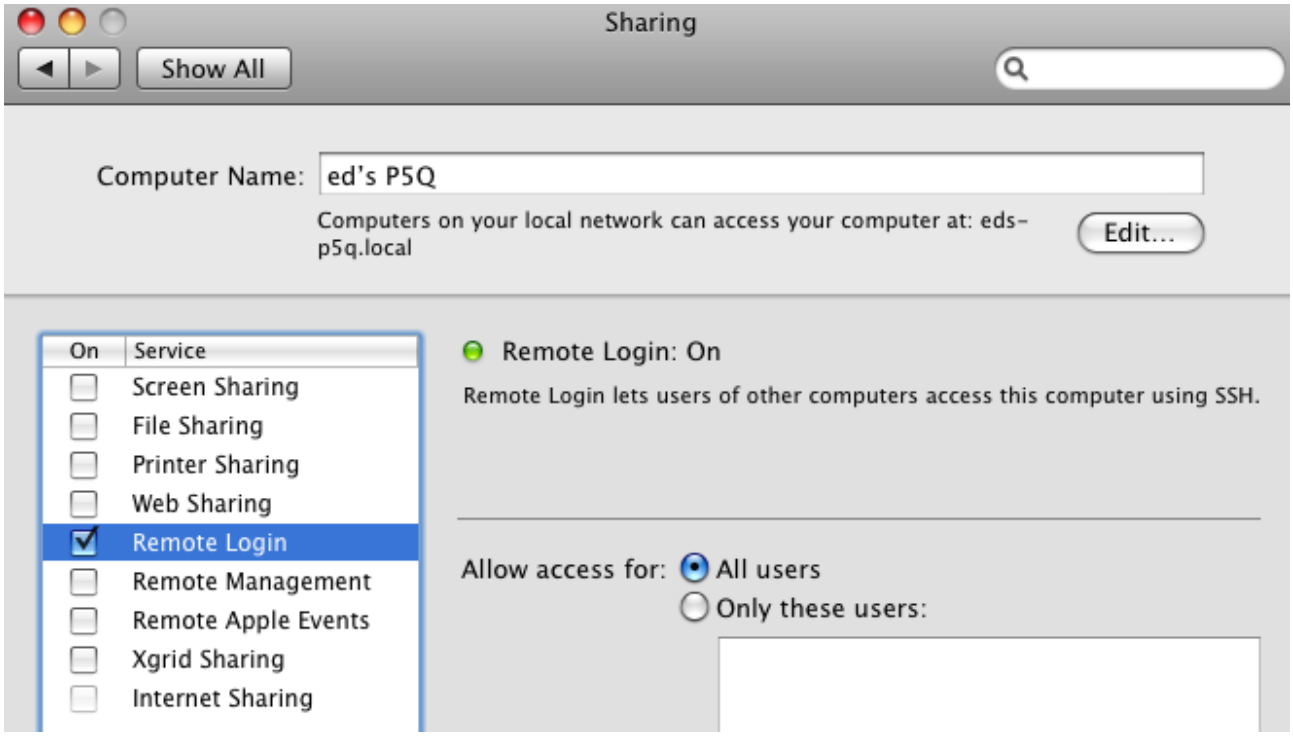
Prezentacja sesji VNC

Sesja SSH

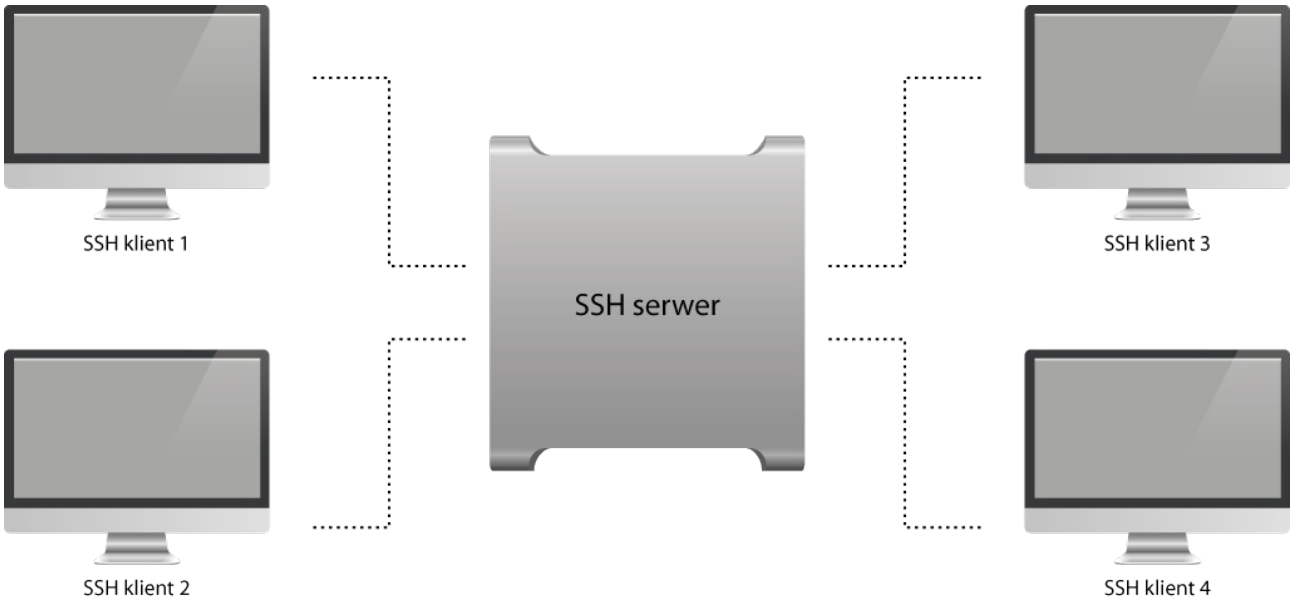
Do komunikacji między komputerami służy wiele protokołów i narzędzi. W tym przykładzie posłużymy się SSH (ang). Zalet SSH jest bardzo dużo, wśród nich m in. szyfrowana transmisja danych. Do zdalnej pracy potrzebujemy serwera, który przez definicję (ang. serve, czyli służyć) będzie udostępniał pewne zasoby. W przypadku SSH będzie to … serwer SSH.

Extra: W przypadku UNIX/-pochodnych systemów serwery SSH mogą nazywać się inaczej (np. dla Ubuntu jest to OpenSSH), a ich administracja przebiega za pomocą innych narzędzi.

Systemy OSX z reguły mają wbudowany serwer SSH, a dostęp do niego (oczywiście posiadając prawa administracyjne) można znaleźć stosując poniższą ścieżkę:
Ścieżka: Logo Apple (lewy, górny róg) > System Preferences/Ustawienia systemowe > Sharing/Udostępnianie > Remote Login/Zdalny dostęp



Po włączeniu serwera SSH możemy mieć do czynienia z sytuacją jak na poniższym diagramie.



Ryc. 2. Diagram połączeń klientów z serwerem SSH.

Warto podkreślić, że od lat 60-tych XX w. ideą budowania tego typu układów było podłączanie do superkomputera (a więc bardzo wydajnej, ale drogiej maszyny) terminali (komputerów o niskiej mocy obliczeniowej, ale za to dość tanich). W efekcie logując się na serwer SSH możemy korzystać z jego zasobów: programów, pamięci, mocy obliczeniowej. Sprawdźmy więc jak to działa!

Dostępny w pracowni serwer SSH ma adres lokalny `10.0.1.124` oraz utworzone konto dla użytkownika `student`. Jeżeli więc pracujesz na maszynie, która ma adres przydzielony wewnątrz tej sieci (tzn. `10.0.1.0-255`) możesz łączyć się z serwerem SSH. Jak sprawdzisz swój adres z poziomu linii komend?

Połączenie SSH wykonamy za pomocą komendy:
```
ssh student@10.0.1.124
```
Następnie zostaniesz poproszony o wprowadzenie hasła:
```
Password:
```

Tu dla celów szkoleniowych hasło jest takie samo jak nazwa użytkownika. Wprowadź hasło. Po prawidłowym zalogowaniu powita nas system.

Extra: Pamiętaj jednak, że w codziennej praktyce nazwa użytkownika (tzw. login) i hasło muszą być różne! Dodatkowo hasło nie powinno być 'słownikowe', tzn. nie powinno być wyrazem umieszczonym w powszechnym słowniku. 'Podłączanie słowników' jest jedną z metod włamań. W praktyce najlepiej, gdy hasło zawiera litery małe i duże oraz cyfry. Im dłuższe hasło tym dłuższy czas potrzebny do jego złamania.

LaTeX – składnia i schemat pracy

Extra: Pozostań zalogowany za serwerze SSH, ew. zaloguj się korzystając ze wskazówek w podrozdziale Sesja SSH.

Przyjrzyjmy się zawartości katalogu domowego użytkownika `student`. Jest tu m. in. katalog `latex_sources`, który jak wskazuje nazwa zawiera pliki źródłowe. Wejdź do tego katalogu i utwórz w nim katalog o nazwie `latex_twojeimię_twojenazwisko` (np. `latex_krzysztof_moszczynski`). Następnie skopiuj do swojego katalogu plik `latex_00.`, który jest umieszczony w katalogu `latex_sources`.
Za pomocą vim zapoznaj się z zawartością skopiowanego pliku:
```
\documentclass{article}

% kodowanie: latin2, utf8 lub cp1250
\usepackage[utf8]{inputenc}

%treść
\begin{document}
To jest pierwszy tekst, jeszcze bez polskich znakow diakrytycznych.
\end{document}
```

Zgodnie ze specyfikacją języka przedstawioną w 'Nie za krótkie wprowadzenie do systemu LATEX 2ε', tłum. Przechlewski T., Kubiak R., Gołdasz J., Rozdz. 1 (praca własna) ten dokument można wytłumaczyć jako:
artykuł, który używa kodowania utf-8 i zawiera podstawową treść, a także dwa komentarze.
Zmodyfikuj dowolnie treść dokumentu, nie używaj jednak polskich znaków diakrytycznych, zapisz plik i wyjdź z trybu edycji vim.

Zadanie 8

1. Za pomocą komendy latex i na podstawie pliku latex_00 wygeneruj plik latex_00.dvi
2. Przekonwertuj plik latex_00.dvi do pliku latex_00.pdf

Podpowiedź: W toku pracy powinniśmy otrzymać podobną zawartość katalogu (oczywiście pliki mogą różnić się rozmiarem, właścicielem, uprawnieniami):
```
eds-p5q:latex_sources ed$ ls -lh
total 72
-rwxrwxrwx  1 ed  staff   205B Feb  8 19:27 latex_00
-rwxrwxrwx  1 ed  staff     8B Feb  8 19:27 latex_00.aux
-rwxrwxrwx  1 ed  staff   300B Feb  8 19:27 latex_00.dvi
-rwxrwxrwx  1 ed  staff    12K Feb  8 19:27 latex_00.log
-rwxrwxrwx  1 ed  staff   6.1K Feb  8 19:27 latex_00.pdf
```

Podsumowując uzyskaliśmy pliki wynikowe i dokonaliśmy pierwszego automatycznego składu. Co

więcej, do tej pracy zostały wykorzystane zasoby serwera i całość jest zdeponowana właśnie na dysku serwera. Aby dokładnie przeglądnąć efekt pracy potrzebujemy przeglądarki plików pdf, a więc środowiska graficznego. Dlatego też wszystkie pliki warto przenieść tym razem na dysk komputera, z którego zalogowałeś się na serwer SSH (tzw. maszyna lokalna, klient).

Zadanie 9

Utwórz archiwum o nazwie `latex_00` i rozszerzeniu/formacie `.tar.gz` lub `.tgz`. Archiwum powinno zawierać wszystkie pliki `latex_00*`.

Zadanie 10

1. Sprawdź absolutną ścieżką utworzonego archiwum.
2. Otwórz nowe okno terminala na maszynie lokalnej i przejdź do swojego katalogu `latex_*`.
3. Utwórz w katalogu `zadania` katalog `10` (porównaj ze strukturą z Ćwiczenie 11).
4. Zmień bieżący katalog na `latex_*/zadania/10`
5. Czas skopiować plik archiwum utworzonego w Zadaniu 9. Do tego celu wykonaj komendę:
   `scp student@10.0.1.124:/ścieżka/absolutna/do/twojego/archiwum ./`
   Podpowiedź: `/ścieżka/absolutna/do/twojego/archiwum -` patrz pkt. 1
6. Wprowadź hasło.
   Podpowiedź: O ile wszystko przebiegło pomyślnie powinniśmy otrzymać raport komendy `scp`:
   ```
   latex_00.tgz            100%  1427     0.4KB/s   00:01
   ```
7. Wypakuj archiwum do bieżącej lokalizacji.
8. Za pomocą dowolnej przeglądarki przeglądnij utworzony plik latex_00.pdf.
9. Zakończ sesję SSH. W tym celu w terminalu, w którym jesteś zalogowany na serwerze wpisz `exit`.

W toku wszystkich ćwiczeń i zadań twój katalog latex_* powinien mieć następującą strukturę:

```
|__
| |__cwiczenia
| | |__11
| | | |__latex_krzysztof_moszczynski.tgz
| |__pliki
| | |__all.txt
| | |__all_1.txt
| | |__plik1.txt
| | |__plik2.txt
| | |__plik3.txt
| | |__plik4.txt
| |__zadania
| | |__03
| | | |__zawartosc_katalogu
| | |__06
| | | |__komenda_tar_gzip
| | |__07
| | | |__plik_vim
| | |__10
| | | |__latex_00
| | | |__latex_00.aux
| | | |__latex_00.dvi
| | | |__latex_00.log
| | | |__latex_00.pdf
```

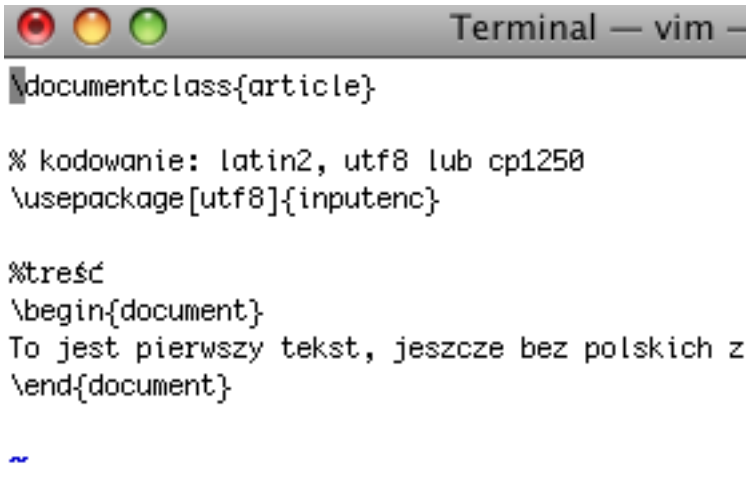
LaTeX i vim

Jeżeli pracujesz na własnej maszynie możesz pobrać odpowiedni dla systemu operacyjnego pakiet dostępny pod adresem: . W przypadku platformy Apple będzie to MacTeX.pkg, dla Linux instalacja może być oparta o uniwersalny pakiet lub zależna od dystrybucji (np. dla Ubuntu jest to pakiet livetext lub livetext-full).
W pracowni stacje robocze zostały wyposażone już w środowisko LaTeX. Dlatego też możliwe jest nie tylko działanie zdalne i lokalny skład tekstów.

Ustawienia vim

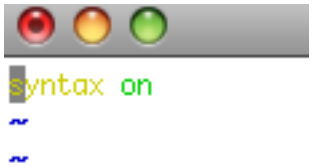
Wykonaj komendę:
`> ~/.vimrc`
która doprowadzi do powstania pliku (ukrytego, jego nazwa rozpoczyna się kropką) ustawień edytora vim.

Przy pomocy vim wejdź w tryb edycji pliku `latex_*/zadania/10/latex_00`.



Mój edytor nie podświetla odpowiednimi kolorami składni przeglądanego pliku. W tym celu wyjdź z edycji pliku `latex_00` i wejdź w tryb edycji pliku `~/.vimrc`. Dodaj w tym pliku następującą frazę:
`syntax on`

Zapisz zmiany, opuść tryb edycji `~/.vimrc`, a następnie ponownie wejdź w tryb edycji tego pliku, aby sprawdzić, czy vim podświetla składnię.



Skoro wszystko przebiegło prawidłowo, opuść tryb edycji `~/.vimrc` i wejdź w tryb edycji `latex_00`. Wciąż brak sukcesów? Sprawdźmy w pierwszej kolejności poleceniem `file`, czy plik na którym operujemy ma odpowiedni format:
```
file latex_krzysztof_moszczynski/zadania/10/latex_00
latex_krzysztof_moszczynski/zadania/10/latex_00: LaTeX 2e document text
```

Skoro jak wyżej wszystko jest w porządku, to wnioskuję, że vim nie potrafi 'domyślić się' w jakim języku jest składnia pliku. Aby pomóc, zmieńmy nazwę pliku na `latex_00.tex`. W końcu udało się:

Extra: Plik konfiguracyjny `.vimrc` może zawierać bardzo dużo ustawień/deklaracji. W przypadku, gdy pracujmy na plikach utf-8 (jest to dziś powszechny rekomendowany sposób kodowania) w pliku można umieścić dodatkową deklarację, tak aby całość wyglądała jak niżej:

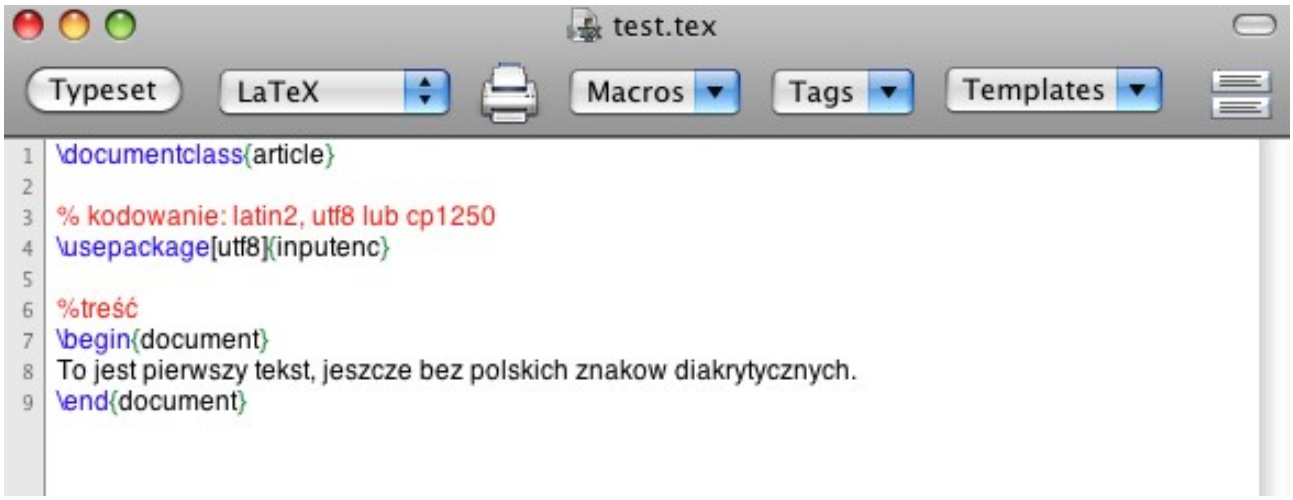
```
set encoding=utf-8
syntax on
```

Edytory GUI

Jest wiele programów, które potrafią zapisywać/eksportować efekt pracy do formatu tex, w tym 'rodzina' pakietów LibreOffice (znane jako StarOffice, OpenOffice lub NeoOffice, w zależności od czasu produkcji i platformy). Jakkolwiek warto podkreślić, że ta funkcjonalność jest tylko dodatkiem i nie czyni tych pakietów dedykowanymi dla LaTeX.
W przypadku instalacji pakietu MacTeX.pkg dostępne są dwa dedykowane edytory: TeXShop, TeXworks. Zaletą używania edytorów dedykowanych jest zautomatyzowanie pracy, np. poprzez korzystanie ze zdefiniowanych rozwiązań lub, co na początku okaże się już dogodniejsze, skrócenie drogi generowania dokumentów do składu (zamiast plików dvi podawany jest rezultat w postaci pdf).

Wprowadzenie do edytora TeXShop

1. W Aplikacjach (Cmd+Shift+A) znajdź katalog TeX, a następnie uruchom TexShop.
2. Wklej źródło pliku latex_00.tex.



3. Zachowując powyższe ustawienia zatwierdź przyciskiem Typeset.
4. Zapisz plik na Biurku jako test.tex zachowując format kodowania jako Unicode UTF-8.

Przestawiony przebieg sesji testowej jest stosunkowo prosty. Poza rezultatem w formacie pdf automatycznie wyświetlanym w przeglądarce, uruchamiane jest również okno konsoli błędów (jest to wbudowany intereface tekstowy LaTeX), z którego to okna można odczytać historię komunikatów generowanych plików, ew. zapis błędów. Do przełączania w sesji między oknami TexShop służy Cmd+~.
Rozważmy przykład błędu w pliku wejściowym: celowo zamieniłem `\begin` na `\begi`. Proces generowania zostaje zatrzymany, a w oknie konsoli błędów wyświetlona jest informacja o nr. wiersza napotkanego błędu oraz sam błąd:

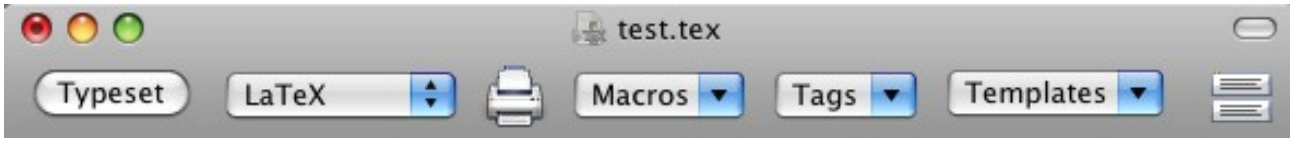
O ile nie wiem co w tej sytuacji zrobić, wystarczy wprowadzić znak zapytania i zatwierdzić klawiszem Return:

```
? ?
Type <return> to proceed, S to scroll future error messages,
R to run without stopping, Q to run quietly,
I to insert something, E to edit your file,
1 or ... or 9 to ignore the next 1 to 9 tokens of input,
H for help, X to quit.
?
```

Wtedy też wyświetli się menu wyboru, którego dokonujemy wprowadzając w polu tekstowym (na dole okna dialogowego):

- `return` – dalszy przebieg
- `S` – przewijanie do kolejnych napotkanych błędów
- `R` – dalszy przebieg, bez dalszego zatwierdzania
- `Q` – dalszy przebieg, bez raportowania ew. błędów
- `I` – wstawienie, przydatne w przypadku poprawy raportowanego błędu
- `E` – powrót do edycji pliku wejściowego
- `1...9` – ignorowanie kolejnych tokenów pliku wejściowego
- `H` – pomoc
- `X` – zakończenie
- ? – wyświetlanie bieżącego komunikatu pomocy

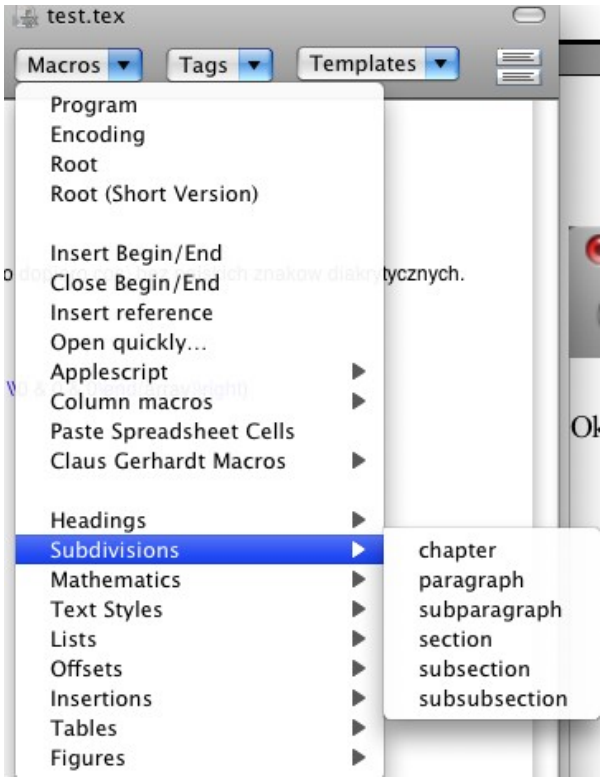
W przypadku analizowanego przez nas pliku (gdzie błędem jest niedokończona składnia: `\begi`), wybierz I, a następnie wprowadź w polu tekstowym `\begin` i zatwierdź klawiszem Return. Błąd został naprawiony, a dzięki temu otrzymaliśmy poprawnie wygenerowany plik pdf.



Okno TeXShop zawiera (od lewej):

- przycisk Typeset, czyli zatwierdzenie procesu składu;
- listę generatorów składu (np. LaTeX, BibTeX itd.);
- możliwość drukowania kodu źródłowego;
- Macros, czyli listę makr (przy pomocy vim możesz tę listę dowolnie modyfikować: `~/Library/TeXShop/Macros/Macros_Latex.plist`) – więcej poniżej;
- Tags, listę znaczników, które łatwo pozwalają poruszać się po kodzie (znaczniki odczytywane są przez program i definiowane z kodu na podstawie wyrażeń LaTeX, np. `\section`, `\subsection` itp.);
- Templates, listę szablonów dla poszczególnych typów dokumentów (dostępne w lokalizacji: `~/Library/TeXShop/Templates/`) – więcej rozdział Szablony;
- przycisk podziału kodu na 2 panele.

TeXShop zawiera wiele rozwiązań udogodnień jak choćby podkreślenie błędów składni, czy autouzupełnienia komend (dostępne po wpisaniu kilku pierwszych liter i zatwierdzeniu klawisza Esc). Ponadto w menu Macros, można wybrać zdefiniowane już komendy:

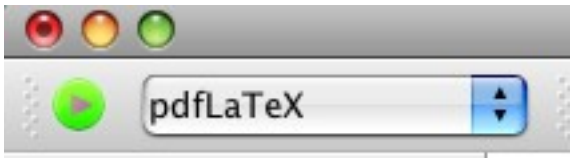


Aby skorzystać z pozostałych, przygotowanych już zasobów zasobów sprawdź menu Window > LaTeX Panel oraz Window > Matrix Panel.

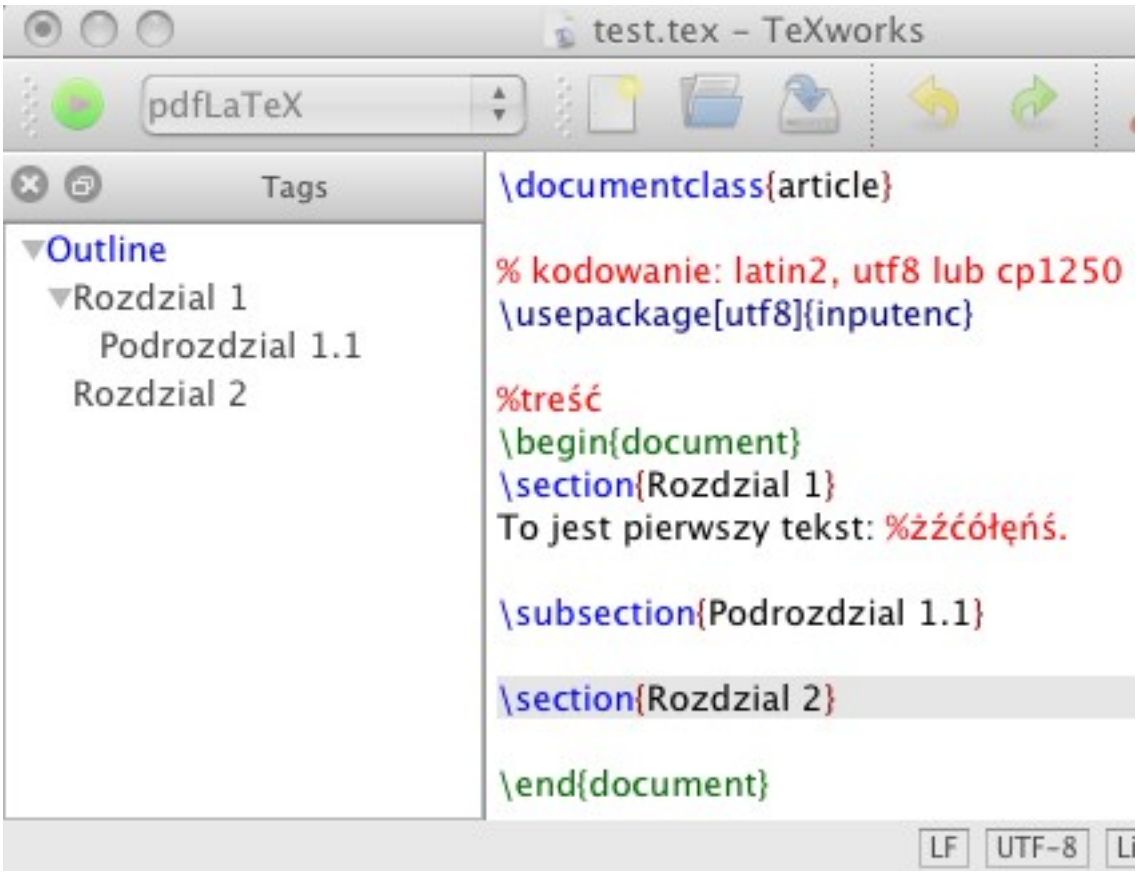
O ile zdecydujesz się na TeXShop jako edytora do dalszej pracy, więcej informacji na temat jego funkcjonalności pod adresem: `http://pages.uoregon.edu/koch/Documentation.pdf`.

Wprowadzenie do edytora TeXworks

Ideą twórców TeXworks było stworzenie prostego narzędzia, które umożliwiałoby komfort pracy porównywalny z edytorami typu WYSIWYG. Mimo że TeXworks jest prostszy niż omawiany wyżej TeXShop, dzięki wbudowanym funkcjonalnością jest nie mniej atrakcyjny.

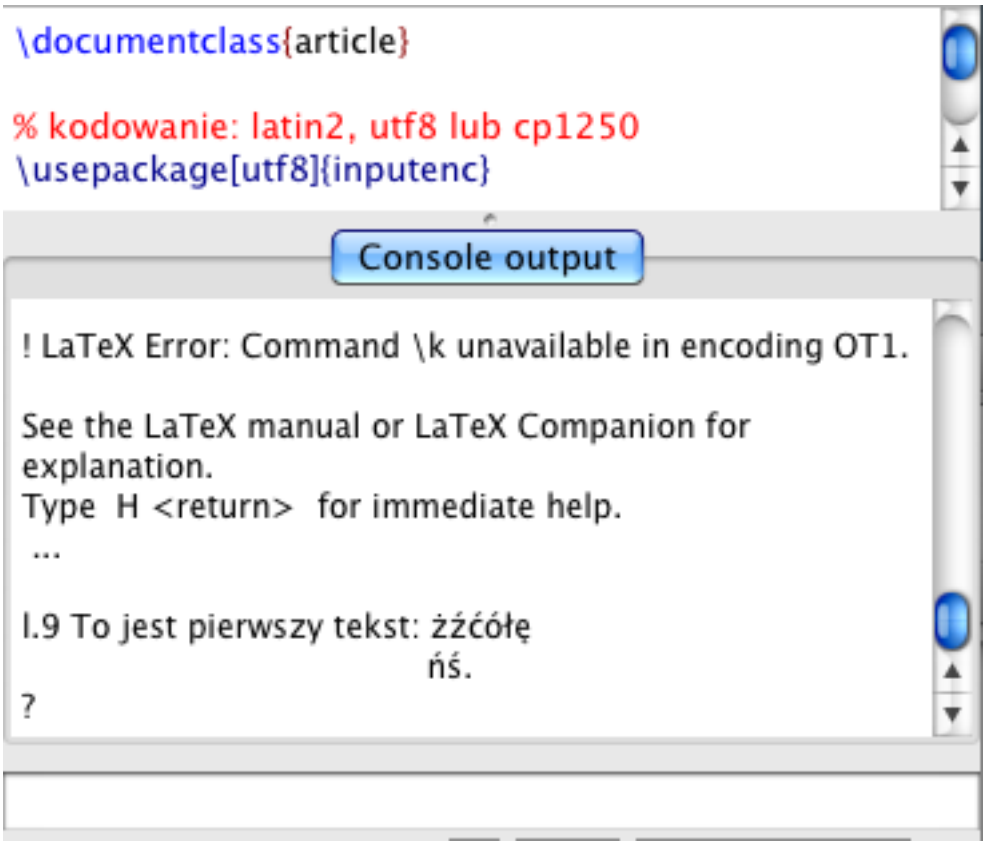


Pasek narzędzi głównego okna TeXworks zawiera po lewej stronie zielony przycisk Typeset, do przeprowadzenia procesu składu. Skład następuje wg wybranego generatora, dla przykładu pdfLaTeX.
Gdy wciśniemy zielony przycisk Typeset zostanie uruchomione dodatkowe okno podglądu rezultatu składu. Ikony narzędzi dla obu okien programu są standardowe, a ich nazwy 'samotłumaczace', w związku z tym nie zostaną tu omówione.
TeXworks jest również wyposażony w podświetlanie składni/błędów oraz autouzupełnianie poleceń (dostępne po naciśnięciu klawisza Tab). Panel dla znaczników dostępny jest po zatwierdzeniu pozycji z górnego menu Window > Show > Tags:



Zmodyfikuj plik latex_00 jak na powyższym zrzucie ekranu, odkomentuj ciąg polskich znaków diakrytycznych i zatwierdź skład tekstu:

W procesie składu wystąpił błąd, który jest oznaczony w lewym górnym rogu czerwoną ikoną. Ponadto została



uruchomiona znajoma już konsola błędów. W tym wypadku komunikat napotkanego błędu informuje, że nie ma możliwości na zrealizowanie składu przy zastosowanym kodowaniu OT1.

Zadanie 11
Na podstawie podręcznika 'Nie za krótkie wprowadzenie …' rozwiąż powyższy problem. Zapisz efekt swojej pacy jako:
`latex_*/zadania/11/latex_01.tex`

O ile zdecydujesz się na TeXworks jako edytora do dalszej pracy, więcej informacji na temat jego funkcjonalności pod adresem: `http://ftp.ctex.org/pub/tex/tools/editors/TeXworks/manual.pdf`.

Dalsza praca

Wybór edytora (np. vim, TexShop, TeXworks, itd.) do dalszej pracy jest kwestią twojej decyzji. Przedstawione poniżej przykłady zostały zrealizowane za pomocą konsoli i edytora vim.

Zadanie 12
Zmodyfikuj dokument `latex_01.tex`, tak aby w treści (między `\begin{document}` a `\end{document}`) zawierał jedynie poniższa frazę, zaczerpniętą z podręcznika 'Nie za krótkie wprowadzenie …':
'W pierwszej części tego rozdziału przedstawimy krótko filozofię oraz historię systemu.'
Na podstawie zdobytej wiedzy i doświadczenia pracy z konsolą błędów znajdź rozwiązanie problemu, który wystąpi podczas składu. Zapisz efekt swojej pacy jako:
`latex_*/zadania/12/latex_02.tex`