

Ćwiczenie 2

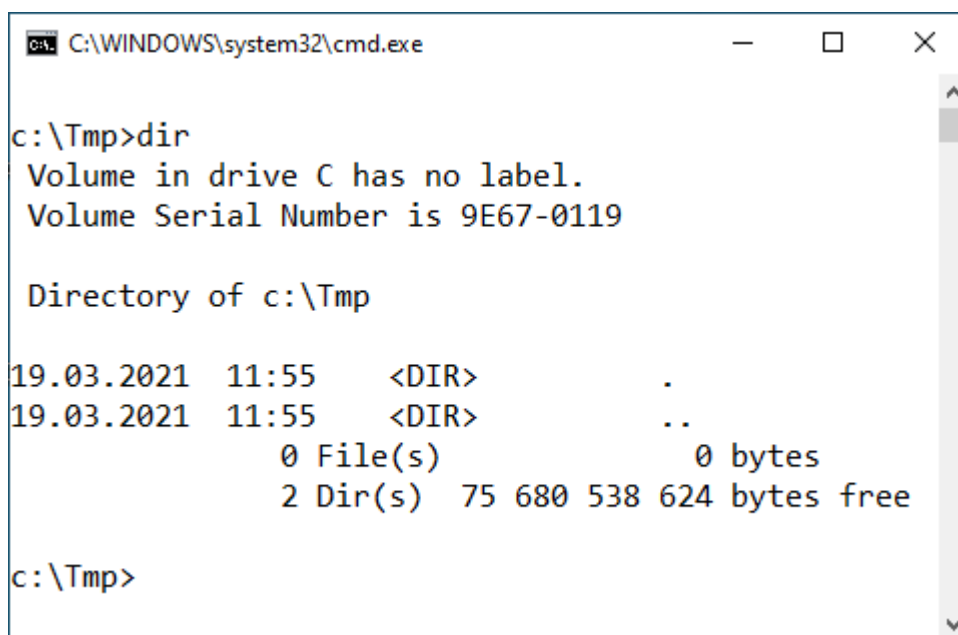
Najprostsze skrypty dla Windows i w bash-u

Cel ćwiczenia:

Przedmiotem ćwiczenia jest przedstawienie możliwości napisania prostych skryptów w środowisku MS Windows i bash-a.

Przebieg ćwiczenia - zagadnienia:

1. Środowisko MS Windows:
 - a. zakładanie katalogów – polecenie md
 - b. przejście do zadanego katalogu: polecenie cd
 - c. kasowanie katalogów – polecenie rd
 - d. zawartość bieżącego katalogu: znaczenie . oraz ..



```
CA. C:\WINDOWS\system32\cmd.exe

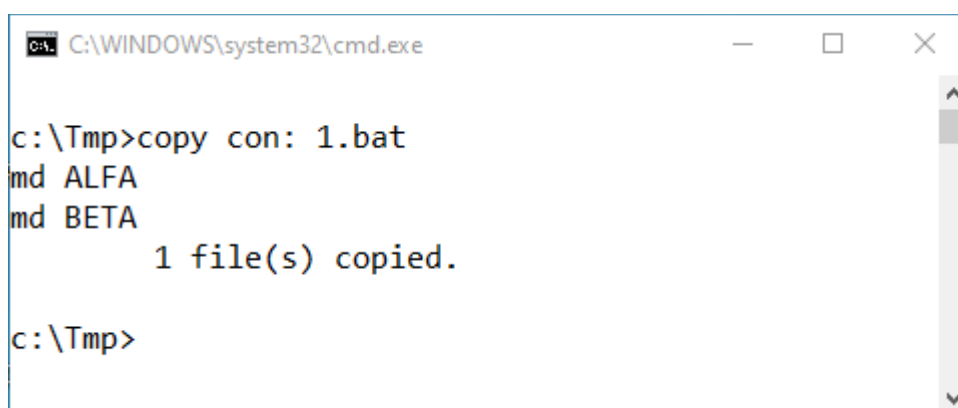
c:\Tmp>dir
Volume in drive C has no label.
Volume Serial Number is 9E67-0119

Directory of c:\Tmp

19.03.2021  11:55    <DIR>          .
19.03.2021  11:55    <DIR>          ..
               0 File(s)                0 bytes
               2 Dir(s)  75 680 538 624 bytes free

c:\Tmp>
```

- e. napisanie prostego skryptu:

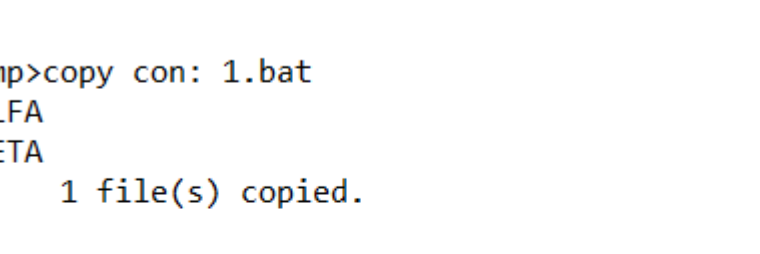


```
CA. C:\WINDOWS\system32\cmd.exe

c:\Tmp>copy con: 1.bat
md ALFA
md BETA
          1 file(s) copied.

c:\Tmp>
```

- f. uruchomienie skryptu:



The screenshot shows a Windows Command Prompt window with the title bar "C:\WINDOWS\system32\cmd.exe". The command history is as follows:

```
c:\Tmp>copy con: 1.bat
md ALFA
md BETA
1 file(s) copied.

c:\Tmp>1.bat

c:\Tmp>md ALFA

c:\Tmp>md BETA

c:\Tmp>
```

2. Dlaczego używamy shell'a:
 - a. co zrobić z wieloma plikami
 - b. wielokrotne wykonywanie tych samych zadań

```
psm@karas: ~  
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 4.4.0-19041-Microsoft x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:       https://ubuntu.com/advantage  
  
System information as of Fri Mar 26 08:09:38 CET 2021  
  
System load:        0.52           Users logged in:      0  
Usage of /home:     unknown       IPv4 address for eth0: 10.0.0.40  
Memory usage:       31%           IPv4 address for eth1: 192.168.17.1  
Swap usage:         0%            IPv4 address for eth2: 192.168.220.1  
Processes:          7  
  
0 updates can be installed immediately.  
0 of these updates are security updates.  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
  
This message is shown once a day. To disable it please create the  
/home/psm/.hushlogin file.  
psm@karas:~$
```

3. Mamy różne Shell'e systemu Linux: jednym z nich jest bash – Bourne Again Shell
- | | | | | | |
|-------------|---|-------|-----------|-------|----------|
| a. Składnia | polecenia: | monit | polecenie | opcje | argument |
| | | \$ | ls | -l | doc |
| b. | Wyświetlenie zawartości bieżącego katalogu: ls -l | | | | |

```

psm@karas: ~
psm@karas:~$ ls -l
total 0
drwxr-xr-x 1 psm psm 512 Mar 24 20:44 alfa
-rw-r--r-- 1 psm psm  6 Mar 24 20:44 beta
psm@karas:~$

```

Znaczenie kolumn:

Numer(y) kolejnego/ych znaku/ów	Interpretacja	Wartość	Interpretacja
1	typ pliku	-	plik
		d	katalog
		l	dowiązanie symboliczne
		p	potok nazwany
		c	urządzenie znakowe
		b	urządzenie blokowe
		s	gniazdo
2-4	uprawnienia właściciela pliku do odczytu (r), zapisu (w) oraz wykonania (x)	r	read
		w	write
		x	executable
5-7	uprawnienia grupy do odczytu (r), zapisu (w) oraz wykonania (x)	r	read
		w	write
		x	executable
8-10	uprawnienia wszystkich użytkowników do odczytu (r), zapisu (w) oraz wykonania (x)	r	read
		w	write
		x	executable
	liczba powiązań do tego elementu		
	ID właściciela pliku		
	ID grupy przypisanej do tego pliku		
	rozmiar		liczba bajtów
	data ostatniej modyfikacji		
	nazwa elementu (pliku)		

- c. zmiana właściciela: chown
- d. zmiana grupy: chgrp
- e. zmiana uprawnień: chmod

przykłady:

```

chmod u+wr plik
chmod g-xr plik
chmod o+w plik
chmod a+r-x plik
chmod 644 plik
chmod 755 plik

```

4. Utworzenie najprostszego pliku:

```
psm@karas: ~  
psm@karas:~$ cat > beta  
ls -l  
^C  
psm@karas:~$
```

5. Wyświetlenie zawartości pliku:

```
psm@karas: ~  
psm@karas:~$ cat > beta  
ls -l  
^C  
psm@karas:~$ cat beta  
ls -l  
psm@karas:~$
```

6. Uruchomienie skryptu

- a. ustawienie atrybutu executable
- b. uruchomienie skryptu:
 - i. ./skrypt
 - ii. bash -x skrypt nie wymaga ustawienia flagi -x

```
psm@karas: ~  
ls -l  
psm@karas:~$ ls -l  
total 0  
drwxr-xr-x 1 psm psm 512 Mar 24 20:44 alfa  
-rw-r--r-- 1 psm psm  6 Mar 24 21:18 beta  
psm@karas:~$ chmod +x beta  
psm@karas:~$ ls -l  
total 0  
drwxr-xr-x 1 psm psm 512 Mar 24 20:44 alfa  
-rwxr-xr-x 1 psm psm  6 Mar 24 21:18 beta  
psm@karas:~$ ./beta  
total 0  
drwxr-xr-x 1 psm psm 512 Mar 24 20:44 alfa  
-rwxr-xr-x 1 psm psm  6 Mar 24 21:18 beta  
psm@karas:~$
```

7. Midnight Commander - mc

- a. wybór edytora (F4)
- b. korekta ustawień wyboru edytora:
update-alternatives --config editor
- c. jeśli nie zadziała to uruchomić:
select-editor
- d. bezpośrednie wywołanie: mcedit

8. Podstawowe polecenie edytora vi

- a. dwa tryby pracy: poleceń i edycji – przełączanie pomiędzy nimi z pomocą klawisza <Esc>
- b. wstawianie tekstu:
 - i. A – dodanie na końcu wiersza
 - ii. I – wstawia na początku bieżącego wiersza
- c. poruszanie kursorem:
 - i. klawisz h – ruch w lewo

- ii. klawisz j – ruch w dół
 - iii. klawisz k – ruch w górę
 - iv. klawisz l – ruch w prawo
 - d. kasowanie znaku x
 - e. kasowanie linii dd
 - f. wycofanie efektu ostatniego polecenia u
 - g. wyjście: <Esc> :wq!
 - h. Inne
 - i. 0 – skok na początek wiersza
 - ii. \$ - skok na koniec wiersza
 - iii. dw – delete word
9. Podstawowe polecenia związane z katalogami:
- a. zakładanie katalogu mkdir
 - b. kasowanie katalogu rmdir
10. Stworzenie najprostszego skryptu do zakładania katalogów

```
psm@karas: ~  
psm@karas:~$ cat > przyklad1  
mkdir ALFA  
mkdir BETA  
^C  
psm@karas:~$ chmod +x przyklad1  
psm@karas:~$ ./przyklad1  
psm@karas:~$ ls -l  
total 0  
drwxr-xr-x 1 psm psm 512 Apr  1 17:28 ALFA  
drwxr-xr-x 1 psm psm 512 Apr  1 17:28 BETA  
-rwxr-xr-x 1 psm psm  22 Apr  1 17:27 przyklad1  
psm@karas:~$
```

11. Stworzenie najprostszego pliku do kasowania katalogów

```
psm@karas: ~  
psm@karas:~$ cat > przyklad2  
rmdir ALFA  
rmdir BETA  
^C  
psm@karas:~$ ls -l  
total 0  
drwxr-xr-x 1 psm psm 512 Apr  1 17:28 ALFA  
drwxr-xr-x 1 psm psm 512 Apr  1 17:28 BETA  
-rwxr-xr-x 1 psm psm  22 Apr  1 17:27 przyklad1  
-rw-r--r-- 1 psm psm  22 Apr  1 17:28 przyklad2  
psm@karas:~$ chmod +x przyklad2  
psm@karas:~$ ./przyklad2  
psm@karas:~$ ls -l  
total 0  
-rwxr-xr-x 1 psm psm  22 Apr  1 17:27 przyklad1  
-rwxr-xr-x 1 psm psm  22 Apr  1 17:28 przyklad2  
psm@karas:~$
```

12. Uzmiennienie nazw katalogów w wyżej wymienionych skryptach poprzez odwołanie do parametrów wywołania skryptu \$1 \$2 \$3

a. zakładanie katalogów

```
psm@karas: ~  
psm@karas:~$ cat > przyklad3  
mkdir $1  
mkdir $2  
^C  
psm@karas:~$ chmod +x przyklad3  
psm@karas:~$ ls -l  
total 0  
-rwxr-xr-x 1 psm psm 22 Apr  1 17:27 przyklad1  
-rwxr-xr-x 1 psm psm 22 Apr  1 17:28 przyklad2  
-rwxr-xr-x 1 psm psm 18 Apr  1 17:31 przyklad3  
psm@karas:~$ ./przyklad3 GAMMA TETA  
psm@karas:~$ ls -l  
total 0  
drwxr-xr-x 1 psm psm 512 Apr  1 17:31 GAMMA  
drwxr-xr-x 1 psm psm 512 Apr  1 17:31 TETA  
-rwxr-xr-x 1 psm psm  22 Apr  1 17:27 przyklad1  
-rwxr-xr-x 1 psm psm  22 Apr  1 17:28 przyklad2  
-rwxr-xr-x 1 psm psm  18 Apr  1 17:31 przyklad3  
psm@karas:~$
```

b. kasowanie katalogów

```
psm@karas: ~  
psm@karas:~$ cat > przyklad4  
rmdir $1  
rmdir $2  
^C  
psm@karas:~$ chmod +x przyklad4  
psm@karas:~$ ls -l  
total 0  
drwxr-xr-x 1 psm psm 512 Apr  1 17:31 GAMMA  
drwxr-xr-x 1 psm psm 512 Apr  1 17:31 TETA  
-rwxr-xr-x 1 psm psm  22 Apr  1 17:27 przyklad1  
-rwxr-xr-x 1 psm psm  22 Apr  1 17:28 przyklad2  
-rwxr-xr-x 1 psm psm  18 Apr  1 17:31 przyklad3  
-rwxr-xr-x 1 psm psm  18 Apr  1 17:32 przyklad4  
psm@karas:~$ ./przyklad4 GAMMA TETA  
psm@karas:~$ ls -l  
total 0  
-rwxr-xr-x 1 psm psm 22 Apr  1 17:27 przyklad1  
-rwxr-xr-x 1 psm psm 22 Apr  1 17:28 przyklad2  
-rwxr-xr-x 1 psm psm 18 Apr  1 17:31 przyklad3  
-rwxr-xr-x 1 psm psm 18 Apr  1 17:32 przyklad4  
psm@karas:~$
```

13. Uruchomienie najprostszego skryptu – przypomnienie:

a. piszemy tekst skryptu:

```
cat > skrypt1  
date  
<Ctrl C>
```

-
- A screenshot of a terminal window. The title bar at the top shows a red icon with a white 'p' and the text 'psm@karas: ~'. The terminal area is mostly empty, with a vertical line of blue tilde characters (~) on the left side. At the bottom, the prompt '"test" [New File]' is visible, followed by '0,0-1' and 'All' on the right.

- ```
psm@karas: ~
psm@karas:~$ ps -ef
UID PID PPID C STIME TTY TIME CMD
root 1 0 0 08:09 ? 00:00:00 /init
root 8 1 0 08:09 tty1 00:00:00 /init
psm 9 8 0 08:09 tty1 00:00:00 -bash
psm 191 9 0 08:18 tty1 00:00:00 vi test
root 192 1 0 08:18 tty2 00:00:00 /init
psm 193 192 0 08:18 tty2 00:00:00 -bash
psm 206 193 0 08:18 tty2 00:00:00 ps -ef
psm@karas:~$
```

- str. 7 z 17

```
psm@karas: ~
psm@karas:~$ kill 191 -9
psm@karas:~$
```

i. efekt „zabicia” programu

```
psm@karas: ~
psm@karas:~$ ps -ef
UID PID PPID C STIME TTY TIME CMD
root 1 0 0 08:09 ? 00:00:00 /init
root 8 1 0 08:09 tty1 00:00:00 /init
psm 9 8 0 08:09 tty1 00:00:00 -bash
psm 91 9 0 08:11 tty1 00:00:00 mc
psm 93 91 0 08:11 pts/0 00:00:00 bash -rcfile .bashrc
psm 99 93 0 08:11 pts/0 00:00:00 ps -ef
Vim: Caught deadly signal TERM
Vim: Finished.

Terminated
psm@karas:~$
```

15. Przykład liczenia znaków, słów, linii:

j. Utwórzmy plik ala.txt o zawartości jak poniżej (wykorzystując np. polecenie `cat > ala.txt`):

```
psm@karas: ~
psm@karas:~$ cat > ala.txt
Pierwsza linijka
Druga linijka
Trzecia linijka
Czwarta linijka
^C
psm@karas:~$
```

k. zobaczymy jak wygląda ten plik i sprawdzimy odpowiednio liczbę znaków, słów, linii:



```
psm@karas: ~
psm@karas:~$ cat ala.txt
Pierwsza linijka
Druga linijka
Trzecia linijka
Czwarta linijka
psm@karas:~$ wc -c ala.txt
63 ala.txt
psm@karas:~$ wc -w ala.txt
8 ala.txt
psm@karas:~$ wc -l ala.txt
4 ala.txt
psm@karas:~$
```

1. jak widać opcje odpowiednio -C -w -l polecenia wc dają możliwość policzenia odpowiednio znaków, słów, linii.

#### 16. Przykład przetwarzania potokowego – liczba studentów na liście

- m. utwórzmy plik lista o zawartości jak poniżej:

```
psm@karas: ~
psm@karas:~$ cat > lista
Babacki
Dabacki
Tabacki
Rabacki
Wabacki
Tabacki
Gabacki
Tabacki
Zabacki
Rabacki
^C
psm@karas:~$
```

- n. wyświetlmy zawartość pliku lista

```
psm@karas: ~
psm@karas:~$ cat lista
Babacki
Dabacki
Tabacki
Rabacki
Wabacki
Tabacki
Gabacki
Tabacki
Zabacki
Rabacki
psm@karas:~$
```

- o. policzmy liczbę linii czyli liczbę nazwisk w tym pliku:

```
psm@karas: ~
psm@karas:~$ cat lista
Babacki
Dabacki
Tabacki
Rabacki
Wabacki
Tabacki
Gabacki
Tabacki
Zabacki
Rabacki
psm@karas:~$ wc -l lista
10 lista
psm@karas:~$
```

- p. ale widać, że nazwiska się powtarzają, czyli otrzymana liczba studentów nie jest poprawna. Aby otrzymać poprawny wynik najpierw posortujmy alfabetycznie nazwiska:

```
psm@karas: ~
psm@karas:~$ sort lista
Babacki
Dabacki
Gabacki
Rabacki
Rabacki
Tabacki
Tabacki
Tabacki
Wabacki
Zabacki
psm@karas:~$
```

- q. wykorzystując konstrukcję przetwarzania potokowego wyeliminujmy powtarzające się nazwiska:

```
psm@karas: ~
psm@karas:~$ sort lista | uniq
Babacki
Dabacki
Gabacki
Rabacki
Tabacki
Wabacki
Zabacki
psm@karas:~$
```

- r. A teraz policzmy liczbę nazwisk – linijek:

```
psm@karas: ~
psm@karas:~$ sort lista | uniq | wc -l
7
psm@karas:~$
```

s. jak widać na liście mamy nazwiska 7 studentów.

17. Wynik instrukcji jako część polecenia.

t. Napiszmy jak poniżej:

```
psm@karas: ~
psm@karas:~$ `echo p``echo wd`
/home/psm
psm@karas:~$ pwd
/home/psm
psm@karas:~$
```

u. Program echo wypisuje na standardowe wyjście treść polecenia w kawałkach. W tym celu poleceni musi się znajdować w odwrotnych apostrofach (klawisz po lewej od klawisza „1”).

v. „Normalny” apostrof oraz podwójny apostrof służą do przekazywania parametrów zawierających wewnątrz spacje. Przykłady:

```
psm@karas: ~
psm@karas:~$ mkdir Ala ma kota
psm@karas:~$ ls -l
total 0
drwxr-xr-x 1 psm psm 512 Mar 26 09:13 Ala
drwxr-xr-x 1 psm psm 512 Mar 26 09:13 kota
drwxr-xr-x 1 psm psm 512 Mar 26 09:13 ma
psm@karas:~$ mkdir "Ala ma kota"
psm@karas:~$ ls -l
total 0
drwxr-xr-x 1 psm psm 512 Mar 26 09:13 Ala
drwxr-xr-x 1 psm psm 512 Mar 26 09:13 'Ala ma kota'
drwxr-xr-x 1 psm psm 512 Mar 26 09:13 kota
drwxr-xr-x 1 psm psm 512 Mar 26 09:13 ma
psm@karas:~$
```

18. Argumenty wywołania skryptu – przykład

```
skrypt1.2 — Notatnik
Plik Edycja Format Widok Pomoc
Przykład na odwołanie do argumentów i ich iteracje
Wywołanie: ./skrypt1.2 Ala ma kota
echo Program $0 wywołano z $# argumentami:
echo pierwszy: $1
echo drugi: $2
echo trzeci: $3
echo wszystkie: $@
shift 2
echo Program $0 wywołano z $# argumentami:
echo pierwszy: $1
echo drugi: $2
echo trzeci:$3
echo wszystkie: $@
```

<uzupełnić>

#### 19. Przykłady wykorzystania znaków:

w. znak: \*                    dowolny ciąg znaków

```
psm@karas: ~
-rw-r--r-- 1 psm psm 63 Mar 27 13:37 ala.txt
-rw-r--r-- 1 psm psm 80 Mar 26 14:20 wynik1
-rw-r--r-- 1 psm psm 80 Mar 26 14:45 wynik11
-rw-r--r-- 1 psm psm 80 Mar 26 14:45 wynik1a1
-rw-r--r-- 1 psm psm 80 Mar 26 14:46 wynik1a2
-rw-r--r-- 1 psm psm 56 Mar 26 14:22 wynik2
-rw-r--r-- 1 psm psm 9 Mar 26 14:22 wynik3
psm@karas:~$ ls -l wynik*
-rw-r--r-- 1 psm psm 80 Mar 26 14:20 wynik1
-rw-r--r-- 1 psm psm 80 Mar 26 14:45 wynik11
-rw-r--r-- 1 psm psm 80 Mar 26 14:45 wynik1a1
-rw-r--r-- 1 psm psm 80 Mar 26 14:46 wynik1a2
-rw-r--r-- 1 psm psm 56 Mar 26 14:22 wynik2
-rw-r--r-- 1 psm psm 9 Mar 26 14:22 wynik3
psm@karas:~$
```

x. znak: ?                    dowolny pojedynczy znak

```
psm@karas: ~
psm@karas:~$ ls -l
total 0
-rw-r--r-- 1 psm psm 63 Mar 27 13:37 ala.txt
-rw-r--r-- 1 psm psm 80 Mar 26 14:20 wynik1
-rw-r--r-- 1 psm psm 80 Mar 26 14:45 wynik11
-rw-r--r-- 1 psm psm 80 Mar 26 14:45 wynik1a1
-rw-r--r-- 1 psm psm 80 Mar 26 14:46 wynik1a2
-rw-r--r-- 1 psm psm 56 Mar 26 14:22 wynik2
-rw-r--r-- 1 psm psm 9 Mar 26 14:22 wynik3
psm@karas:~$ ls -l wynik?
-rw-r--r-- 1 psm psm 80 Mar 26 14:20 wynik1
-rw-r--r-- 1 psm psm 56 Mar 26 14:22 wynik2
-rw-r--r-- 1 psm psm 9 Mar 26 14:22 wynik3
psm@karas:~$ ls -l wynik1?1
-rw-r--r-- 1 psm psm 80 Mar 26 14:45 wynik1a1
psm@karas:~$
```

y. może być więcej znaków ?

```
psm@karas: ~
psm@karas:~$ ls -l
total 0
-rw-r--r-- 1 psm psm 63 Mar 27 13:37 ala.txt
-rw-r--r-- 1 psm psm 80 Mar 26 14:20 wynik1
-rw-r--r-- 1 psm psm 80 Mar 26 14:45 wynik11
-rw-r--r-- 1 psm psm 80 Mar 26 14:45 wynik1a1
-rw-r--r-- 1 psm psm 80 Mar 26 14:46 wynik1a2
-rw-r--r-- 1 psm psm 56 Mar 26 14:22 wynik2
-rw-r--r-- 1 psm psm 9 Mar 26 14:22 wynik3
psm@karas:~$ ls -l wynik?a?
-rw-r--r-- 1 psm psm 80 Mar 26 14:45 wynik1a1
-rw-r--r-- 1 psm psm 80 Mar 26 14:46 wynik1a2
psm@karas:~$
```

20. Określenie liczby znaków w parametrze wywołania skryptu

z. treść skryptu

```
skrypt1.3 — Notatnik
Plik Edycja Format Widok Pomoc
Skrypt okresla liczbe znakow w parametrze wywolania
tego skryptu.
case $1 in
 ?) echo $1 ma jeden znak ;;
 ??) echo $1 ma dwa znaki ;;
 ???) echo $1 ma trzy znaki ;;
 *) echo $1 na wiecej niz trzy znaki ;;
esac
```

aa. wynik działania skryptu

```
psm@karas: ~
psm@karas:~$./skrypt3 a
a ma jeden znak
psm@karas:~$./skrypt3 al
al ma dwa znaki
psm@karas:~$./skrypt3 ala
ala ma trzy znaki
psm@karas:~$./skrypt3 adam
adam ma wiecej niz trzy znaki
psm@karas:~$ █
```

21. Odczytanie standardowego wejścia:  
bb. Wydajmy następujące polecenie:

```
psm@karas: ~
psm@karas:~$ read a
wartosc
psm@karas:~$ echo $a
wartosc
psm@karas:~$ read a b c
Ala ma kota
psm@karas:~$ echo $a $b $c
Ala ma kota
psm@karas:~$
```

- cc. Wczytywanie danych z wyświetleniem zachęty:

```
psm@karas: ~
psm@karas:~$ cat skrypt2
echo -n "Podaj nazwę:"
read nazwa
echo "Moja nazwa to: " $nazwa
psm@karas:~$./skrypt2
Podaj nazwę:tresc
Moja nazwa to: tresc
psm@karas:~$
```

- dd. opcja -n w poleceniu echo powoduje brak przejścia kursora do nowej linii.  
22. Nazwa zmiennej i wartość zmiennej:

```
psm@karas: ~
psm@karas:~$ zmienna=wartosc
psm@karas:~$ echo $zmienna
wartosc
psm@karas:~$
```

23. Proste operacje arytmetyczne:  
ee. obliczenie wartości arytmetycznej

```
psm@karas: ~
psm@karas:~$ expr 1 + 3
4
psm@karas:~$ expr 1 - 3
-2
psm@karas:~$ expr 10 / 5
2
psm@karas:~$ expr 10 * 5
expr: syntax error: unexpected argument 'ala.txt'
psm@karas:~$ expr 10 * 5
50
psm@karas:~$
```

ff. podstawienie wartości obliczenia pod zmienną

```
psm@karas: ~
psm@karas:~$ echo $i

psm@karas:~$ i=`expr 1 + 3`
psm@karas:~$ echo $i
4
psm@karas:~$
```

gg. podstawienie wyniku obliczenia na zmiennych pod zmienną

```
psm@karas: ~
psm@karas:~$ i=4
psm@karas:~$ echo $i
4
psm@karas:~$ i=`expr $i + $i + 2`
psm@karas:~$ echo $i
10
psm@karas:~$
```

## 24. Polecenie touch

hh. umożliwia zakładanie plików o długości 0 bajtów

```
psm@karas: ~
psm@karas:~$ touch a1.txt a2.txt a3.txt a1.inny a2.inny
psm@karas:~$ ls -l
total 0
-rw-r--r-- 1 psm psm 0 Mar 28 07:34 a1.inny
-rw-r--r-- 1 psm psm 0 Mar 28 07:34 a1.txt
-rw-r--r-- 1 psm psm 0 Mar 28 07:34 a2.inny
-rw-r--r-- 1 psm psm 0 Mar 28 07:34 a2.txt
-rw-r--r-- 1 psm psm 0 Mar 28 07:34 a3.txt
psm@karas:~$
```

ii. tutaj możemy wyfiltrować pliki po nazwie z użyciem ?

```
psm@karas: ~
total 0
-rw-r--r-- 1 psm psm 0 Mar 28 07:34 a1.inny
-rw-r--r-- 1 psm psm 0 Mar 28 07:34 a1.txt
-rw-r--r-- 1 psm psm 0 Mar 28 07:34 a2.inny
-rw-r--r-- 1 psm psm 0 Mar 28 07:34 a2.txt
-rw-r--r-- 1 psm psm 0 Mar 28 07:34 a3.txt
psm@karas:~$ ls -l a?.txt
-rw-r--r-- 1 psm psm 0 Mar 28 07:34 a1.txt
-rw-r--r-- 1 psm psm 0 Mar 28 07:34 a2.txt
-rw-r--r-- 1 psm psm 0 Mar 28 07:34 a3.txt
psm@karas:~$
```

25. Polecenie if - przykłady:

```
psm@karas: ~
psm@karas:~$ if test 1 == 1 ; then echo PRAWDA ; else echo fałsz ; fi
PRAWDA
psm@karas:~$ if test 1 == 0 ; then echo PRAWDA ; else echo fałsz ; fi
fałsz
psm@karas:~$ if [1 == 1] ; then echo PRAWDA ; else echo fałsz ; fi
PRAWDA
psm@karas:~$ if [1 == 0] ; then echo PRAWDA ; else echo fałsz ; fi
fałsz
psm@karas:~$ _
```

26. Prosty skrypt – porównanie dwóch argumentów.

jj. w skrypcie testowana jest obecność dokładnie dwóch argumentów

```
psm@karas: ~
psm@karas:~$ cat skrypt5
Napisz skrypt, który w zależności od wartości dwóch argumentów
jego wywołania wyświetla następujące wartości:
arg1 < arg2 to wartość: -1
arg1 = arg2 to wartość: 0
arg1 > arg2 to wartość: 1
Przykładowe wywołanie skryptu: ./skrypt5 3 4 daje wartość -1
Skrypt powinien być wyposażony w sprawdzanie liczby argumentów jego wywołania
if test $# -ne 2 ; then
 echo "Niepoprawna liczba argumentów - wywołanie: ./skrypt5 arg1 arg2"
else
 if [$1 -lt $2] ; then
 echo -1
 elif [$1 -eq $2] ; then
 echo 0
 else
 echo 1
 fi
fi
psm@karas:~$
```

kk. Spróbujemy sprawdzić działanie tego skryptu



```
psm@karas: ~
psm@karas:~$./skrypt5
Niepoprawna liczba argumentów - wywołanie: ./skrypt5 arg1 arg 2
psm@karas:~$./skrypt5 3 4 5
Niepoprawna liczba argumentów - wywołanie: ./skrypt5 arg1 arg 2
psm@karas:~$./skrypt5 3 4
-1
psm@karas:~$./skrypt5 3 3
0
psm@karas:~$./skrypt5 4 3
1
psm@karas:~$
```

oczywiście, aby uruchomić ten skrypt, pamiętajmy o atrybucie x – „execut