**KONTAKT**
Adrian Florek
adrian.florek@pwr.edu.pl



## Część Praktyczna

### 1. Zmienne

a) Napisz skrypt wymagający podania imienia i wieku w argumencie, przypisz do zmiennych, a następnie wyświetl te zmienne w formie: „Imie: , Wiek: ”

```
$ hello.sh
1   #!/bin/bash
2   echo "Imie:$1, Wiek:$2"
```

```
┌──(kali㉿kali)-[~/nowykatalog]
└─$ bash hello.sh Dawid 19
Imie:Dawid, Wiek:19
```

b) Utwórz skrypt, który pobiera od użytkownika długość i szerokość prostokąta. Oblicza pole powierzchni prostokąta i wyświetli wynik na ekranie.

```
$ hello.sh
1   #!/bin/bash
2   dlugosc=$1
3   szerokosc=$2
4   Pole_powierzchni=$(echo "$dlugosc * $szerokosc" | bc)
5   echo "Pole powierzchni tego prostokta wynosi ${Pole_powierzchni}"
```

```
└─$ bash hello.sh  10 100
Pole powierzchni tego prostokta wynosi 1000
```

### 2. Instrukcje warunkowe

a) Napisz skrypt który zapyta o nazwę użytkownika i sprawdzi czy dany użytkownik to admin. Jeżeli tak zwróć odpowiedź „Konto admina”, jeżeli nie zwróć „To nie jest konto admina”

```
$ hello2.sh
1   #!/bin/bash
2   echo "Podaj nazwe uzytkownika."
3   read nazwa_uzytkownika
4   if [[ ${nazwa_uzytkownika} == "admin" ]];
5   then
6       echo "konto admina"
7   else
8       echo "To nie jest konto admina"
9   fi
```

**KONTAKT**
Adrian Florek
adrian.florek@pwr.edu.pl

```
┌──(kali㉿kali)-[~/nowykatalog]
└─$ bash hello2.sh
Podaj nazwe uzytkownika.
admin
konto admina

┌──(kali㉿kali)-[~/nowykatalog]
└─$ bash hello2.sh
Podaj nazwe uzytkownika.
kali
To nie jest konto admina
```

b) Napisz skrypt, który zapyta o nazwę pliku, a następnie sprawdzi czy plik istnieje, czy jest pusty, czy jest katalogiem i wypisze odpowiedź na każde pytanie w osobnej linii.

| Operator | Description: Expression True if... |
|---|---|
| !EXPRESSION | The EXPRESSION is false. |
| -n STRING | STRING length is greater than zero |
| -z STRING | The length of STRING is zero (empty) |
| STRING1 != STRING2 | STRING1 is not equal to STRING2 |
| STRING1 = STRING2 | STRING1 is equal to STRING2 |
| INTEGER1 -eq INTEGER2 | INTEGER1 is equal to INTEGER2 |
| INTEGER1 -ne INTEGER2 | INTEGER1 is not equal to INTEGER2 |
| INTEGER1 -gt INTEGER2 | INTEGER1 is greater than INTEGER2 |
| INTEGER1 -lt INTEGER2 | INTEGER1 is less than INTEGER2 |
| INTEGER1 -ge INTEGER2 | INTEGER1 is greater than or equal to INTEGER 2 |
| INTEGER1 -le INTEGER2 | INTEGER1 is less than or equal to INTEGER 2 |
| -d FILE | FILE exists and is a directory |
| -e FILE | FILE exists |
| -r FILE | FILE exists and has read permission |
| -s FILE | FILE exists and it is not empty |
| -w FILE | FILE exists and has write permission |
| -x FILE | FILE exists and has execute permission |

**KONTAKT**
Adrian Florek
adrian.florek@pwr.edu.pl



$ hello3.sh

```
#!/bin/bash
read -p "Podaj nazwe pliku: " plik
if [[ -e $plik ]];
then
    echo "Plik istnieje"
else
    echo "Plik nie istnieje"
fi

if [[ -s $plik ]];
then
    echo "Plik nie jest pusty"
else
    echo "Plik jest pusty"
fi

if [[ -d $plik ]];
then
    echo "Plik jest katalogiem"
else
    echo "Plik nie jest katalogiem"
fi
```

```
┌──(kali㉿kali)-[~/nowykatalog]
└─$ ls
hello2.sh  hello3.sh  hello.sh  nowykatalog1

┌──(kali㉿kali)-[~/nowykatalog]
└─$ bash hello3.sh
Podaj nazwe pliku: hello2.sh
Plik istnieje
Plik nie jest pusty
Plik nie jest katalogiem

┌──(kali㉿kali)-[~/nowykatalog]
└─$ bash hello3.sh
Podaj nazwe pliku: nieistieje
Plik nie istnieje
Plik jest pusty
Plik nie jest katalogiem

┌──(kali㉿kali)-[~/nowykatalog]
└─$ bash hello3.sh
Podaj nazwe pliku: nowykatalog1
Plik istnieje
Plik nie jest pusty
Plik jest katalogiem
```

**KONTAKT**
Adrian Florek
adrian.florek@pwr.edu.pl



## 3. Operatory logiczne

a) Napisz skrypt, który zapyta o liczbę, a następnie przy pomocy operatora AND sprawdzi czy liczba jest dodatnia i czy jest nieparzysta, a na koniec wyświetli wynik

```
$ hello4.sh
1   #!/bin/bash
2   read -p "Podaj liczbę: " liczba
3   if [[ $liczba -gt 0 && $((liczba % 2)) -ne 0 ]];
4   then
5       echo "Liczba jest dodatnia i nieparzysta."
6   else
7       echo "Liczba nie jest dodatnia i nieparzysta."
8   fi
```

```
┌──(kali㉿kali)-[~/nowykatalog]
└─$ bash hello4.sh
Podaj liczbę: 4
Liczba nie jest dodatnia i nieparzysta.

┌──(kali㉿kali)-[~/nowykatalog]
└─$ bash hello4.sh
Podaj liczbę: 5
Liczba jest dodatnia i nieparzysta.
```

b) Napisz skrypt, który poprosi użytkownika o podanie nazwy użytkownika i hasła, a następnie sprawdzi, czy podane dane są poprawne (np. nazwa użytkownika to "admin" i hasło to "password").

```
1   #!/bin/bash
2   read -p "Podaj login: " login
3   read -p "Podaj haslo: " haslo
4   if [[ $haslo == "password" && $login == "admin" ]];
5   then
6       echo "Zalogowano!"
7   else
8       echo "Niepoprawne dane"
9   fi
```

**KONTAKT**
Adrian Florek
adrian.florek@pwr.edu.pl

```
┌──(kali㉿kali)-[~/nowykatalog]
└─$ bash hello4.sh
Podaj login: adam
Podaj haslo: haslo
Niepoprawne dane

┌──(kali㉿kali)-[~/nowykatalog]
└─$ bash hello4.sh
Podaj login: admin
Podaj haslo: password
Zalogowano!
```

## 4. Pętle

a) Napisz skrypt generujący 10 adresów IP za pomocą pętli for, które będą różniły się czwartym oktetem rosnąco w adresie 192.168.1.X

```
$ hello4.sh
1  #!/bin/bash
2  numbers="0 1 2 3 4 5 6 7 8 9"
3  for number in ${numbers}
4  do
5      echo "192.168.1.${number}"
6  done
```

```
┌──(kali㉿kali)-[~/nowykatalog]
└─$ bash hello4.sh
192.168.1.0
192.168.1.1
192.168.1.2
192.168.1.3
192.168.1.4
192.168.1.5
192.168.1.6
192.168.1.7
192.168.1.8
192.168.1.9
```

b) Napisz ten sam skrypt za pomocą pętli while

```
$ hello4.sh
1  #!/bin/bash
2  counter=0
3  while [[ $counter -le 9 ]]
4  do
5      echo 192.168.1.$counter
6      ((counter++))
7  done
```

**KONTAKT**
Adrian Florek
adrian.florek@pwr.edu.pl

```
┌──(kali㉿kali)-[~/nowykatalog]
└─$ bash hello4.sh
192.168.1.0
192.168.1.1
192.168.1.2
192.168.1.3
192.168.1.4
192.168.1.5
192.168.1.6
192.168.1.7
192.168.1.8
192.168.1.9
```

## 5. Funkcje

a) Utwórz skrypt wyświetlający informacje o systemie: Aktualna data, Wersja linuxa, Użytkownik, AdresIP. Każda informacja powinna znajdować się w osobnej funkcji. Użyj jednej zmiennej z nazwą komputera w każdej funkcji.
Przykład: „Aktualna data na (Nazwa komputera) to X”

$ hello5.sh

```
1  #!/bin/bash
2  komputer=$(hostname)
3  echo "Aktualna data na $komputer to $(date)"
4  echo "Wersj linuxa na $komputer to $(uname -r)"
5  echo "Użytkownik na $komputer to $(whoami)"
6  ip_address=$(hostname -I | awk '{print $1}')
7  echo "Adres IP komputera $komputer to $ip_address"
```

```
└─$ bash hello5.sh
Aktualna data na kali to Wed Mar 26 09:41:29 AM EDT 2025
Wersj linuxa na kali to 6.11.2-amd64
Użytkownik na kali to kali
Adres IP komputera kali to 10.0.2.15
```