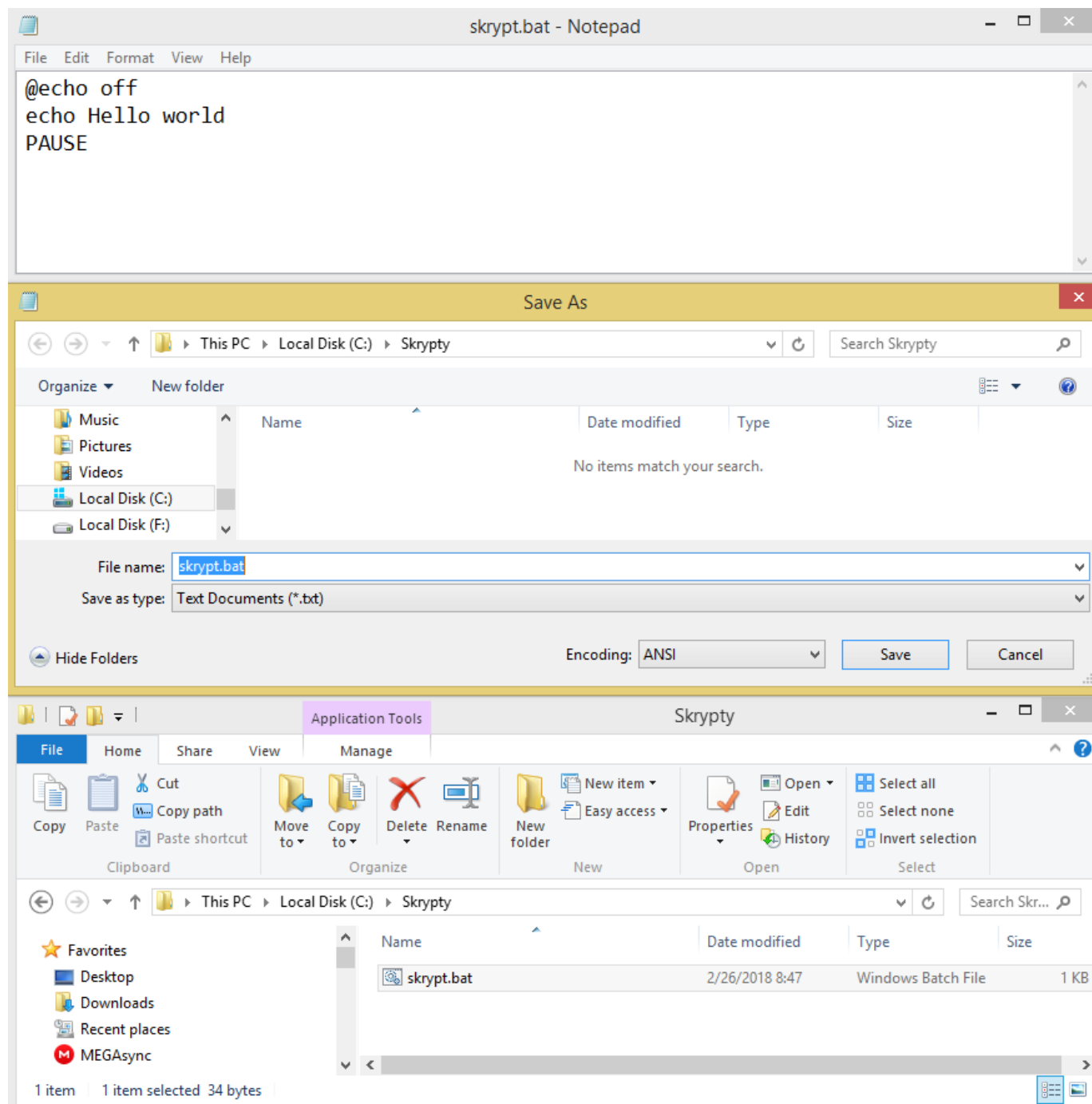


Programowanie wsadowe DOS/Windows – Część I

1. Tworzenie skryptów

Program wsadowy jest to ciąg poleceń (zwany także skrypcem) trybu linii komend lub wywołań programów zapisany w pliku tekstowym o rozszerzeniu **.bat**

Skrypty można tworzyć za pomocą dowolnego edytora tekstu (np. notatnik), zapisując je z dodaniem rozszerzenia **.bat**



Zadanie 1.

Utworzyć skrypt o treści z powyższego zrzutu ekranu i uruchomić (klikając na ikonę skrypt.bat). Usunąć pierwszą linijkę skryptu i spróbować uruchomić ponownie. Następnie usunąć ostatnią linijkę i uruchomić.


2. Wiersz poleceń

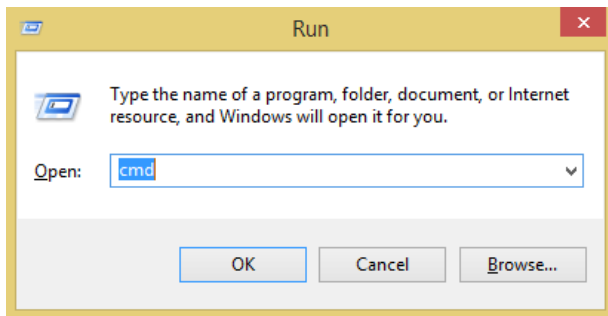
W skryptach można wykorzystywać wszystkie komendy wiersza poleceń (ang. *command line interpreter*). Najważniejsze komendy wiersza poleceń można znaleźć w instrukcjach:

<http://www.iisi.pcz.pl/index.php/pl/do-pobrania?func=fileinfo&id=465> (Windows 1)

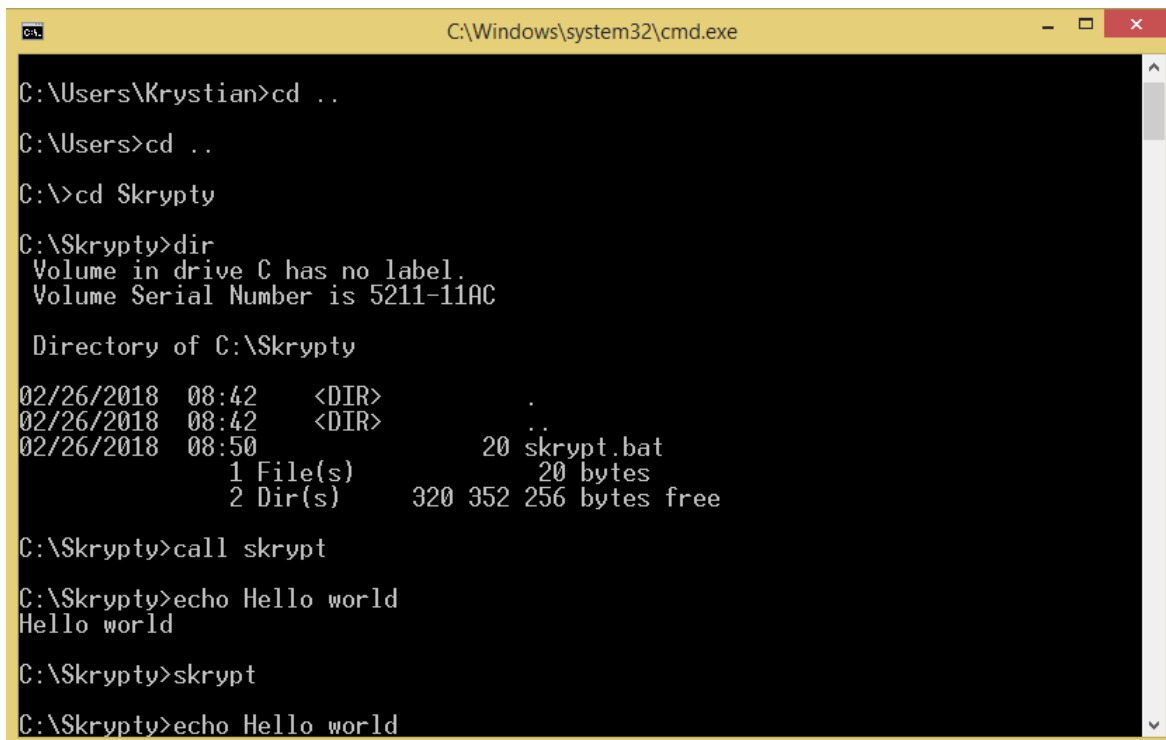
<http://www.iisi.pcz.pl/index.php/pl/do-pobrania?func=fileinfo&id=466> (Windows 2)

<http://www.iisi.pcz.pl/index.php/pl/do-pobrania?func=fileinfo&id=469> (Windows 3)

Wiersz poleceń można uruchomić wciskając kombinację klawiszy  + R i wpisując **cmd**:



Za pomocą komendy **cd** można poruszać się po katalogach, za pomocą komendy **dir** można wyświetlić listę plików (domyślnie z bieżącego katalogu), za pomocą komendy **call plik** lub wpisując **nazwę skryptu** można uruchomić dowolny skrypt:



```
C:\Windows\system32\cmd.exe

C:\Users\Krystian>cd ..
C:\Users>cd ..
C:\>cd Skrypty
C:\Skrypty>dir
Volume in drive C has no label.
Volume Serial Number is 5211-11AC

Directory of C:\Skrypty
02/26/2018  08:42    <DIR>        .
02/26/2018  08:42    <DIR>        ..
02/26/2018  08:50                20 skrypt.bat
               1 File(s)                20 bytes
               2 Dir(s)          320 352 256 bytes free

C:\Skrypty>call skrypt
C:\Skrypty>echo Hello world
Hello world

C:\Skrypty>skrypt
C:\Skrypty>echo Hello world
```

Komenda **cd katalog** otwiera podany katalog, wpisując **cd ..** komenda przechodzi do katalogu nadrzędnego, wpisując **cd /** komenda przechodzi do katalogu głównego.

Zadanie 2.

Uruchomić wiersz poleceń i za jego pomocą wywołać skrypt utworzony w zadaniu 1. Sprawdzić różnice w wywołaniu skryptu wpisując **jego nazwę** i wykorzystując komendę **call**.

3. Podstawowe komendy wiersza poleceń:

Wierszu poleceń umożliwia dostęp do wielu różnych wbudowanych komend (wewnętrznych i zewnętrznych). Szczegółowy opis komend podstawowych można znaleźć w następującej instrukcji:

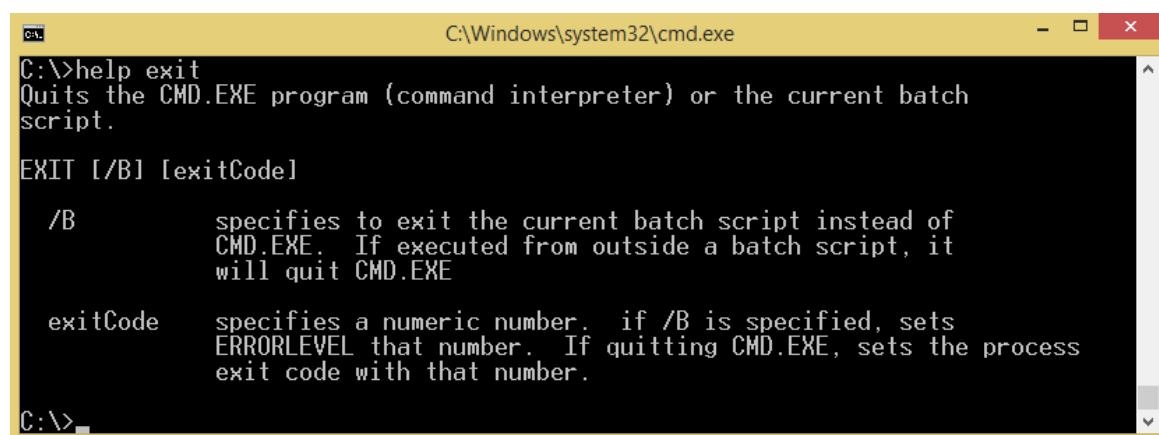
<http://www.iisi.pcz.pl/index.php/pl/do-pobrania?func=fileinfo&id=466>

Do najważniejszych z nich należą (w nawiasach podano nazwę alternatywną):

- time - wyświetla lub ustawia czas systemowy
- date - wyświetla lub ustawia datę systemową zapisaną w CMOS
- exit - kończy pracę wiersza poleceń lub wykonywanego skryptu
- vol - wyświetla etykietę woluminu dysku i numer seryjny, jeżeli istnieją
- chdir (cd) - wyświetla lub zmienia bieżący katalog
- copy - kopiowanie plików z lokalizacji źródłowej do lokalizacji docelowej / łączenie plików
- cls - czyści ekran wiersza poleceń
- del - usuwa pliki
- mkdir (md) - umożliwia założenie nowego katalogu
- rename (ren) - zmienia nazwę jednego lub więcej plików
- rd (rmdir) - umożliwia usuwanie folderów
- type - wyświetla zawartość pliku
- title - zmienia tytuł bieżącego okna konsoli
- prompt - zmienia domyślny tryb zgłoszenia konsoli
- findstr - wyszukuje ciąg znaków w plikach
- subst - przypisuje ścieżkę do nazwy dysku / tworzy dysk wirtualny
- label - wyświetla etykietę woluminu dysku i numer seryjny
- attrib - wyświetla lub zmienia atrybuty plików
- help - wyświetla listę dostępnych komend / wyświetla informacje o komendach

Większość komend umożliwia ich uruchamianie z dodatkowymi parametrami i przełącznikami.

Ich listę i dokładny opis można wyświetlić wykorzystując polecenie **help** *komenda* lub **komenda** /?:



```
C:\Windows\system32\cmd.exe
C:\>help exit
Quits the CMD.EXE program (command interpreter) or the current batch
script.

EXIT [/B] [exitCode]

    /B          specifies to exit the current batch script instead of
                  CMD.EXE.  If executed from outside a batch script, it
                  will quit CMD.EXE

    exitCode    specifies a numeric number.  if /B is specified, sets
                  ERRORLEVEL that number.  If quitting CMD.EXE, sets the process
                  exit code with that number.

C:\>
```

Jak widać polecenie **exit** można uruchomić z **opcjonalnym** (informuje o tym nawias kwadratowy) przełącznikiem /B i **opcjonalnym** parametrem exitCode

Zadanie 3.

Wyświetlić informacje o przedstawionych powyżej komendach. Wywołać komendę **dir** w taki sposób aby wyświetlała za pomocą **małych liter** tylko nazwy plików i folderów **ukrytych** z katalogu C:\Windows.

4. Zmienne

W wierszu poleceń zmienne można definiować za pomocą polecenia **SET** o następującej składni:

SET nazwa=wartość

Uwaga: przed i po znaku = nie może być żadnej **spacji** (wynika to z przyjętej składni skryptów). Aby wyświetlić wartość zmiennej należy ująć nazwę zmiennej w znaki %.

Przykład:

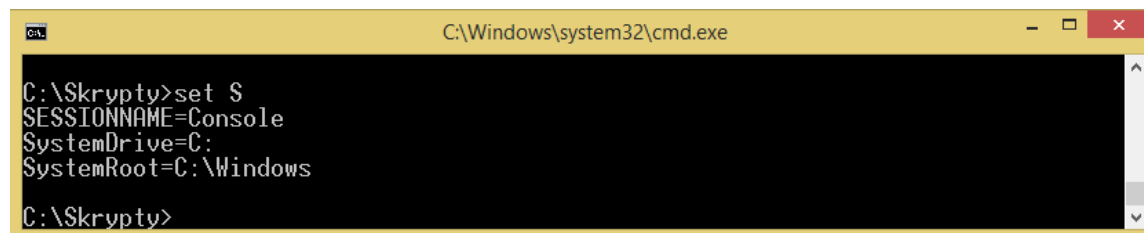
```
@echo off
set zm1 = 10
set zm2=40
set zm3=test
set zm4=drugi test
set zm5="trzeci test"
echo %zm1% - %zm2% - %zm3% - %zm4% - %zm5%
```

W celu **przypisania** do zmiennej wartości wyrażenia (np. arytmetycznego) należy używać przełącznika /A.

Przykład:

```
@echo off
set zm1=5
set zm2=10
set zm3=%zm1%+%zm2%
set /A zm4=%zm1%+%zm2%
echo %zm3% - %zm4%
```

W wierszu poleceń dostępnych jest wiele zmiennych, w tym zmienne **środowiskowe**, w celu ich wyświetlenia wystarczy wpisać komendę **set** (bez żadnych parametrów). W celu wyświetlenia listy zmiennych zaczynających się według określonego ciągu znaków należy użyć komendy **set ciąg**:



```
C:\Windows\system32\cmd.exe

C:\Skrypty>set S
SESSIONNAME=Console
SystemDrive=C:
SystemRoot=C:\Windows
C:\Skrypty>
```

Oprócz zmiennych środowiskowych, dostępne są także zmienne specjalne, np:

- %CD% - bieżący katalog
- %DATE% - bieżąca data
- %TIME% - bieżący czas
- %RANDOM% - wartość losowa z zakresu od 0 do 32767
- %ERRORLEVEL% - wartość błędu (0 gdy nie wystąpił)

Zadanie 4.

Uruchomić skrypty z powyższych przykładów. Zastanowić się nad uzyskanym wynikiem.

Zadanie 5* (rozwiązanie zadań oznaczonych * należy umieścić w sprawozdaniu lub pokazać podczas zajęć).

Napisać skrypt który zmieni kolor konsoli (polecenie **color**), zmieni tytuł okna (polecenie **title**) oraz tryb zgłoszenia (polecenie **prompt**), wyświetli tekst powitania zawierający nazwę użytkownika (należy ją odnaleźć w zmiennych środowiskowych), bieżącą datę oraz jedną liczbę losową.

5. Instrukcje warunkowe

Instrukcje warunkowe przyjmują następującą postać:

- | | |
|--|---|
| • if [not] exist plik polecenie | - sprawdzenie czy plik istnieje |
| • if defined zmienna polecenie | - sprawdza czy jest utworzona dana zmienna |
| • if [not][/] łańcuch1==łańcuch2 polecenie | - porównanie dwóch łańcuchów
/I oznacza brak uwzględnienia wielkości liter |
| • if [/] wartość1 OP wartość2 polecenie | - porównanie dwóch wartości, OP oznacza operator:
EQU – ciągi są równe
NEQ – ciągi są różne
LSS – pierwszy ciąg jest mniejszy niż drugi
LEQ – pierwszy ciąg jest równy lub mniejszy drugiemu
GTR – pierwszy ciąg jest większy niż drugi
GEQ – pierwszy ciąg jest równy lub większy drugiemu |

Przykład:

```
@echo off
mkdir katalog1
if %ERRORLEVEL%==0 echo Brak błędu
if not %ERRORLEVEL%==0 echo Wystąpił błąd: %ERRORLEVEL%
```

Instrukcje warunkowe mogą także przyjąć bardziej rozbudowaną postać:

```
if warunek (
    polecenie1
    polecenie2
    ...
) else (
    polecenie3
    polecenie4
    ...
)
```

Uwaga: nawias (musi znajdować się w tej samej linii co początek polecenia

Przykład:

```
@echo off
mkdir katalog2
if %ERRORLEVEL%==0 (
    echo Brak błędu
) else (
    echo Wystąpił błąd: %ERRORLEVEL%
)
```

Zadanie 6.

Uruchomić kilkakrotnie skrypty z powyższych przykładów. Zastanowić się nad uzyskanym wynikiem.

Zadanie 7* (rozwiązanie zadań oznaczonych * należy umieścić w sprawozdaniu lub pokazać podczas zajęć).

Wykorzystując przełącznik /P polecenia **set** pobrać od użytkownika nazwę katalogu jaki ma zostać utworzony. Jeżeli dany katalog istnieje, zmienić jego nazwę na *nazwa.old* (lub zapytać użytkownika co zrobić – np. czy usunąć dany katalog). Przykład wczytania danych od użytkownika: set /P katalog="Podaj nazwę katalogu: "

6. Instrukcja skoku

Instrukcja skoku służy do ustalenia kolejności wykonywania instrukcji w plikach wsadowych:

goto etykieta

Etykiety można definiować w dowolnym miejscu kodu, poprzedzając je znakiem dwukropka:

:etykieta

Przykład (pominięcie tekstu):

```
@echo off
goto koniec
echo Ten tekst
echo zostanie pominięty
:koniec
echo Ten tekst
echo zostanie wyświetlony
```

Przykład (pętla za pomocą goto):

```
@echo off
set i=0
:powrot
echo i=%i%
set /A i=%i%+1
if %i% LSS 10 goto powrot
```

Zadanie 8.

Napisać skrypt który będzie sumował wartości liczbowe podawane przez użytkownika (po podaniu wartości należy ją doliczyć do sumy). W przypadku podania wartości równej 0 skrypt powinien zakończyć działanie.

7. Argumenty skryptu

Do każdego skryptu można przekazywać argumenty, należy to robić w sposób następujący:

skrypt arg1 arg2 arg3 ...

Dostęp do kolejnych argumentów wewnątrz skryptu jest następujący: %1 %2 %3 ..., ponadto:

- %* - zawiera zestaw wszystkich argumentów w postaci jednego ciągu znaków
- %0 - zawiera nazwę polecenia
- shift - polecenie przesuwa argumenty w lewo (%1 przyjmuje wartość %2, itd.)

Przykład:

```
@echo off
echo Argument 1: %1
echo Argument 2: %2
echo Nazwa skryptu: %0
echo Argumenty: %*
shift
echo Argument 1 po shift: %1
```

Zadanie 9.

Przetestować powyższy kod podając różne zestawy argumentów do skryptu.

8. Pętle

Pętle przyjmują następującą postać:

- for /L %%i in (start, skok, koniec) do polecenie - pętla indeksowana
- for %%i in (zestaw) do polecenie - pętla przechodząca po elementach listy
- for /F [delims=x] %%i in ('komenda ') do polecenie - pętla przechodząca po rezultacie komendy
delims (domyślnie spacja i tabulator) oznacza
miejsce rozdzielenia nazwy z listy

Przykład:

```
@echo off
echo Petla nr.1:
for /L %%i in (1, 1, 5) do echo %%i
echo Petla nr.2:
for %%i in (11 13 17 29) do echo %%i
echo Petla nr.3:
for /F "delims=%TAB%" %%i in ('dir C:\ /B') do echo %%i
```

Uwaga: pętle traktowane są jako jednoetapowe polecenia, w których globalne wartości zmiennych uaktualniane są dopiero po zakończeniu działania pętli. Poniższy kod nie zadziała zgodnie z oczekiwaniami:

Przykład - nieintuicyjne działanie pętli:

```
@echo off
set suma=0
for /L %%i in (1,1,5) do (
    set /A suma=%suma%+%%i
    echo Wartosc: %suma%
)
echo Suma: %suma%
```

Powyższy przykład wyświetli: Suma: 5 (wewnątrz pętli wartość %suma% cały czas wynosi 0). Aby temu zapobiec należy wywołać specjalne polecenie: **setlocal EnableDelayedExpansion** i wartości zmiennych globalnych wewnątrz pętli wywoływać wykorzystując znaki **wykrzyknika** zamiast znaków procentów:

Przykład - działanie pętli – wersja poprawna:

```
setlocal EnableDelayedExpansion
@echo off
set suma=0
for /L %%i in (1,1,5) do (
    set /A suma=!suma!+%%i
    echo Wartosc: !suma!
)
echo Suma: %suma%
```

Zadanie 10.

Napisać skrypt wykonujący pętle wyświetlającą wartości parzyste: 10 8 6 4 2 0.

Zadanie 11. * (rozwiązanie zadań oznaczonych * należy umieścić w sprawozdaniu lub pokazać podczas zajęć).

Napisać pętlę przechodzącą po nazwach wszystkich procesów wyświetlanych za pomocą polecenia **tasklist**. Pętla powinna dodatkowo zliczać wystąpienia procesu o nazwie svchost.exe oraz notepad.exe.

9. Zadania do wykonania

Zadanie 12.

Napisać skrypt oparty na pętli `for` tworzący następujące katalogi: `kat1` `kat3` `kat5` `kat7` `kat9`

Zadanie 13.

Napisać skrypt pobierający od użytkownika 3 zmienne (działanie, `x`, `y`), w zależności od podanego numeru działania skrypt powinien:

- działanie == 1 – zsumować wartość `x` i `y`
- działanie == 2 – odjąć `x` od `y`
- działanie == 3 – odjąć `y` od `x`
- działanie == 4 – pomnożyć `x` przez `y`

Zadanie 14. * (rozwiązanie zadań oznaczonych * należy umieścić w sprawozdaniu lub pokazać podczas zajęć).

Napisać skrypt przyjmujący dwa argumenty:

- liczbę liczb losowych które mają zostać wygenerowane (od 1 do 10)
- maksymalną wartość jaka ma zostać wygenerowana (parametr opcjonalny)

Wyświetlić odpowiednie komunikaty błędów.

Skrypt może zostać oparty zarówno na pętli `for` jak i instrukcji skoku `goto`.

10. Zadania dodatkowe (opcjonalne)

Zadanie 15.

Napisać skrypt wyświetlający informacje o systemie operacyjnym i procesorze (na podstawie zmiennych środowiskowych).

Zadanie 16.

Napisać skrypt zliczający liczbę parametrów podanych jako argumenty.

Rozwiązania części zadań:

Zadanie nr. 8:

```
@echo off
set suma=0
:powrot
set /P dodaj="Podaj wartosc: "
if %dodaj%==0 goto koniec
set /A suma=%suma%+dodaj
echo Wynik=%suma%
goto powrot
:koniec
```

Zadanie nr. 13 (alternatywne rozwiązanie za pomocą goto):

```
@echo off
:poczatek
set /P dzialanie="Wybierz dzialanie: 1=x+y, 2=x-y, 3=y-x, 4=x*y: "
set /P x="Podaj X: "
set /P y="Podaj Y: "
goto dzial%dzialanie%
:dzial1
set /A wynik=%x%+%y%
goto koniec
:dzial2
set /A wynik=%x%-y%
goto koniec
:dzial3
set /A wynik=%y%-x%
goto koniec
:dzial4
set /A wynik=%x%*y%
goto koniec
:koniec
echo Wynik: %wynik%
set /P wybor="Wykonac kolejne dzialanie? T/N: "
if /I "%wybor%"=="t" goto poczatek
```