Windows – Skrypty

Do czego służą skrypty?

Skrypt #1 – wyłączenie komputera po określonym czasie

Skrypt #2 – anulowanie wyłączenia komputera

Skrypt #3 – wylogowanie użytkownika

Skrypt #4 – Tworzenie plików

Skrypt #5 – utworzenie struktury katalogów

Skrypt #6 – zmiana daty systemowej

Skrypt #7 – użycie zmiennych środowiskowych

Skrypt #8 – sortowanie plików

Skrypt #9 – masowa zmiana nazw plików

Skrypt #10 – ukrycie wszystkich plików

Skrypt #11 – usuwanie plików

Skrypt #12 – usuwanie folderów

Skrypt #13 – pętle

Skrypt #14 – zmiana koloru okna co sekundę

Skrypt #15 – zapisanie drzewka dysku C:\ do pliku

Skrypt #16 – modyfikowanie ustawień kont użytkowników

Skrypt #17 – masowe operacje na kontach użytkowników

Skrypt #18 – zapisywanie „wypluć” do pliku i autostart

Do czego służą skrypty?

Skrypty batch służą w systemie Windows głównie zautomatyzowaniu wykonywania niektórych poleceń lub ich ciągów. Zamiast za każdym razem wpisywać polecenia w celu wykonania jakiejś dłuższej akcji, możemy zapisać je do pliku .bat lub .cmd i uruchamiać wielokrotnie i w dowolnym momencie. Kolejne polecenia zapisuje się w nowych wierszach na przykład:

```
@ECHO OFF
ECHO Witaj!
PAUSE
ECHO Ala ma kota.
```

@ECHO OFF to polecenie służące ukrywaniu ścieżki i znaku zachęty po wykonaniu każdego z rozkazów. Zwiększa to przejrzystość okna.

Skrypty takie mają przeróżne zastosowania od wyświetlania tekstu czy parametrów komputera, przez proste instalatory i launchery innych programów, po nawet bardziej skomplikowane, które można by już nazwać programami, na przykład BPaint autorstwa KVC.

Obrazek 1: BPaint – hardcorowy przykład zastosowania skryptów batch

Oczywiście ten skrypt to już mocna przesada – skrypty batch nie służą zwykle do tworzenia aż tak zaawansowanych rzeczy. Poniżej znajduje się kilka przykładów jak możemy wykorzystać programy wsadowe w codziennym życiu.

Skrypt #1 – wyłączenie komputera po określonym czasie

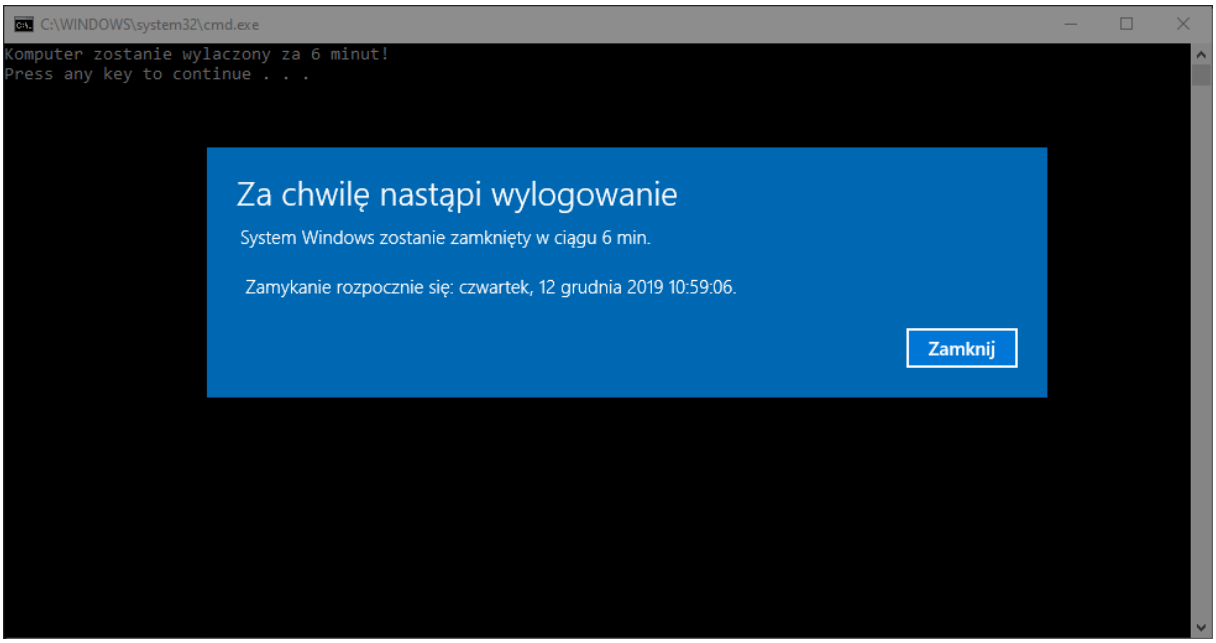
Można do tego wykorzystać zwykłe polecenie shutdown /s z parametrem /t:[liczba sekund].

polecenie1.bat

```
@echo off
shutdown -s -t 360
echo Komputer zostanie wylaczony za 6 minut!
pause
```

Obrazek 2: Skrypt #1

Jak widać, na końcu dodałem informację o wykonanym skrypcie oraz polecenie pause, które poprosi użytkownika o naciśnięcie dowolnego przycisku w celu kontynuowania wykonywania skryptu. Po nim skrypt się kończy, więc okno zostanie po prostu zamknięte.



Obrazek 3: Działanie skryptu #1

Po uruchomieniu skryptu Windows wyświetla nam piękny niebieski komunikat oraz podaje dokładny czas, w którym komputer zostanie wyłączony.

Skrypt #2 – anulowanie wyłączenia komputera

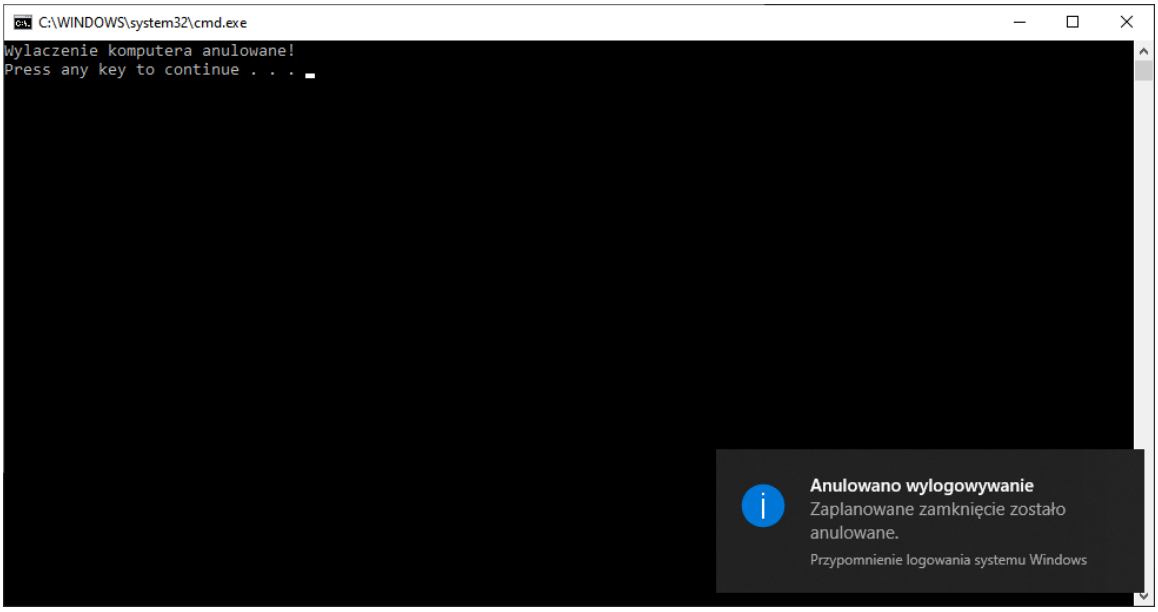
Ten skrypt z kolei jest w stanie zatrzymać działanie poprzedniego. Po jego uruchomieniu będziemy mogli znów cieszyć się naszym uruchomionym systemem bez obawy, że za chwilę zostanie wyłączony. Windows również powiadomi nas o tej akcji, ale tym razem zwykłym powiadomieniem, nie okienkiem.

polecenie2.bat

```
@echo off
shutdown -a
echo Wylaczenie komputera anulowane!
pause
```

Obrazek 4: Skrypt #2

Zostało tu użyte polecenie `shutdown` z parametrem `-a`, który anuluje zamknięcie systemu.



Obrazek 5: Działanie skryptu #2

Skrypt #3 – wylogowanie użytkownika

Znów bardzo podobne polecenie do poprzednich, tym razem jednak użyjemy argumentu `/l`. Z jakiegoś powodu jednak, nie możemy używać go razem z `/t`, więc tym sposobem nie uda nam się wylogować usera po upływie danego czasu.

polecenie3.bat

```
@echo off
echo Uzytkownik zostanie wylogowany!
shutdown -l
pause
```

Obrazek 6: Skrypt #3

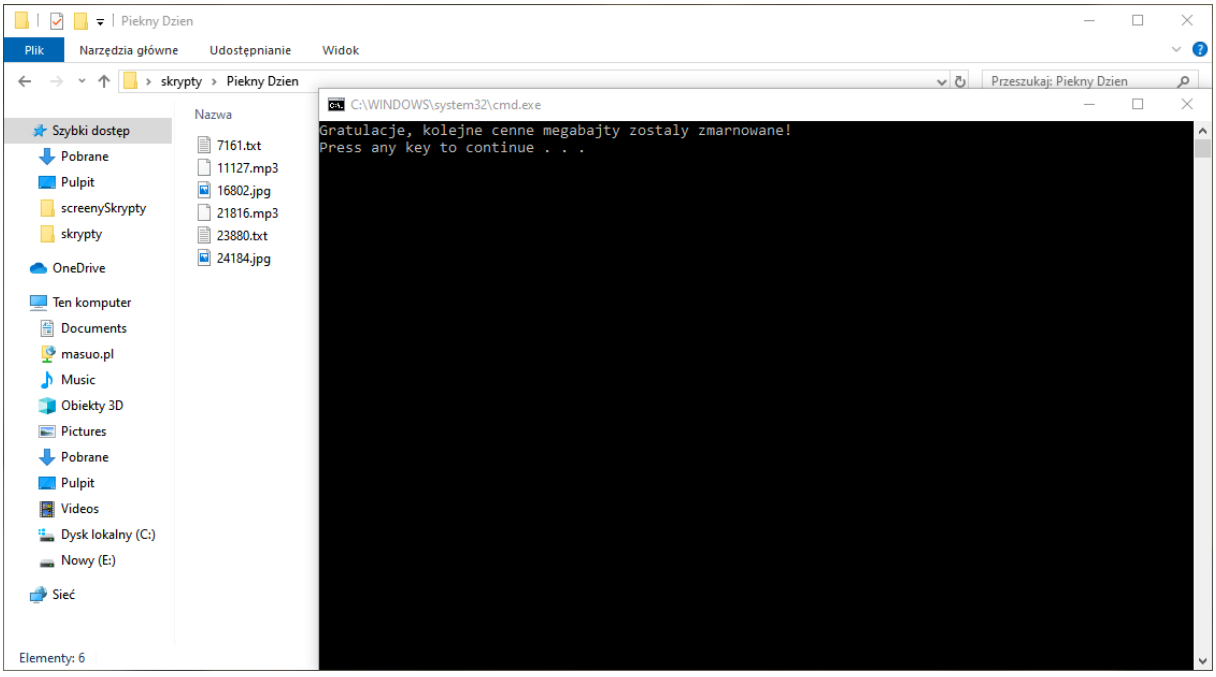
Skrypt #4 – Tworzenie plików

Skrypt ten utworzy nam z każdym jego uruchomieniem 6 śmieciowych plików w folderze „Piekny Dzien”. Będą to po dwie mp3’ki, dwa pliki tekstowe i dwa obrazki. Ten skrypt idealnie ukazuje nam to, jak skrypty mogą ułatwić nam życie. Bez nich trzeba by za każdym razem wpisywać wszystkie te polecenia ręcznie.

polecenie4.bat

```
@echo off
mkdir "Piekny Dzien"
cd "Piekny Dzien"
echo och >> %random%.txt
echo ach >> %random%.txt
echo wow >> %random%.mp3
echo omg >> %random%.mp3
echo pif >> %random%.jpg
echo paf >> %random%.jpg
echo Gratulacje, kolejne cenne megabajty zostaly zmarnowane!
pause
```

Obrazek 7: Skrypt #4



Obrazek 8: Działanie skryptu #4

Jak widać, został utworzony wspomniany folder, a w nim 6 plików-śmieci. Każde kolejne uruchomienie programu spowoduje utworzenie kolejnych 6 plików.

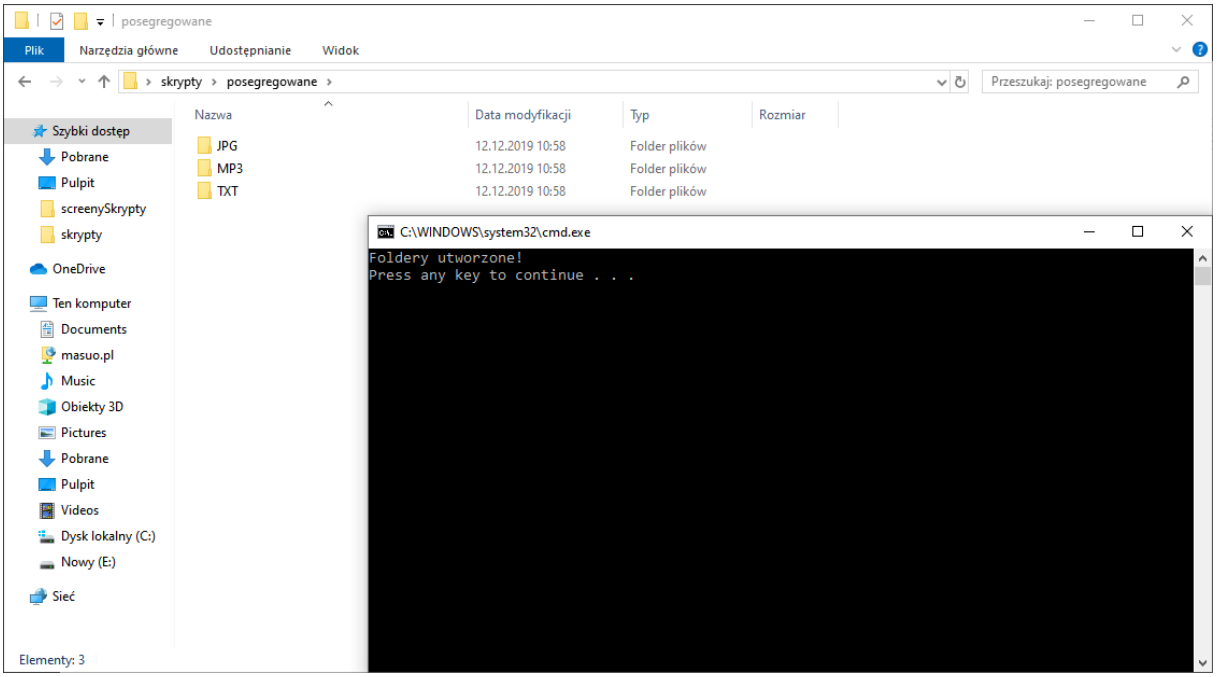
Skrypt #5 – utworzenie struktury katalogów

Kolejny skrypt utworzy nam proste drzewko katalogów, a mianowicie w folderze „posortowane” znajdą się foldery „TXT”, „MP3” oraz „JPG”.

polecenie5.bat

```
@echo off
mkdir posegregowane
cd posegregowane
mkdir TXT MP3 JPG
echo Foldery utworzone!
pause
```

Obrazek 9: Skrypt #5



Obrazek 10: Działanie skryptu #5

Skrypt #6 – zmiana daty systemowej

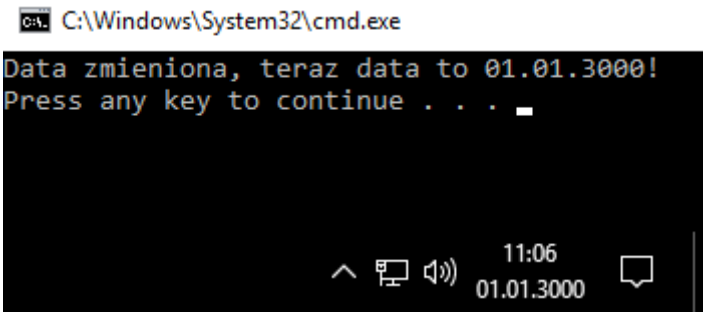
W tym przypadku wykorzystamy polecenie date oraz zmienną środowiskową %date% w celu wyświetlenia daty obecnej.

polecenie6.bat

```
@echo off
date 01-01-3000
echo Data zmieniona, teraz data to %date%!
pause
```

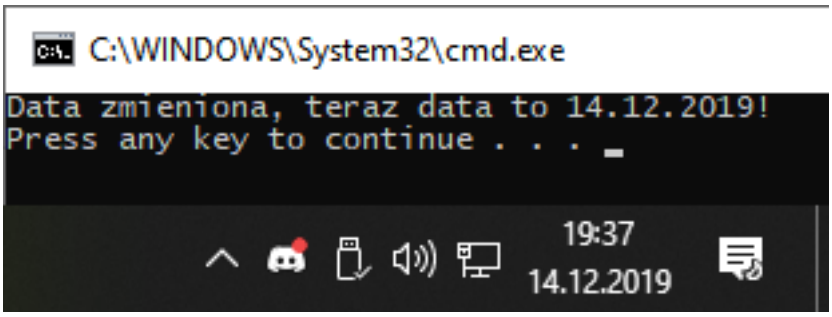
Obrazek 11: Skrypt #6

Aby skrypt się wykonał, należy uruchomić go w trybie administratora



Obrazek 12: Działanie skryptu #6

Aby powrócić do dzisiejszych czasów, Zmieniamy datę na dziś i uruchamiamy ponownie skrypt jako admin.



Obrazek 13: Powrót do teraźniejszości

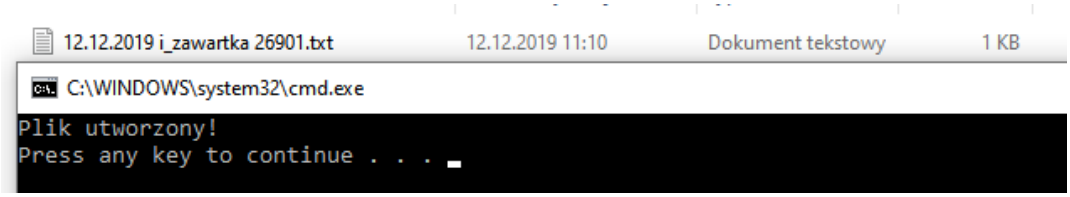
Skrypt #7 – użycie zmiennych środowiskowych

Zmienne środowiskowe to zmienne, których wartość jest ustawiana automatycznie przez system na bieżąco lub przy instalacji programów, uruchamianiu ich itp. Zawierają najważniejsze parametry systemowe, z których mogą korzystać wszystkie programy. Ten skrypt utworzy plik tekstowy, który w swojej nazwie będzie zawierał obecną datę, nazwę użytkownika oraz losową cyfrę.

polecenie7.bat

```
@echo off
echo dzień dobry! >> "%date% %username% %random%.txt"
echo Plik utworzony!
pause
```

Obrazek 14: Skrypt #7



Obrazek 15: Działanie skryptu #7

Skrypt #8 – sortowanie plików

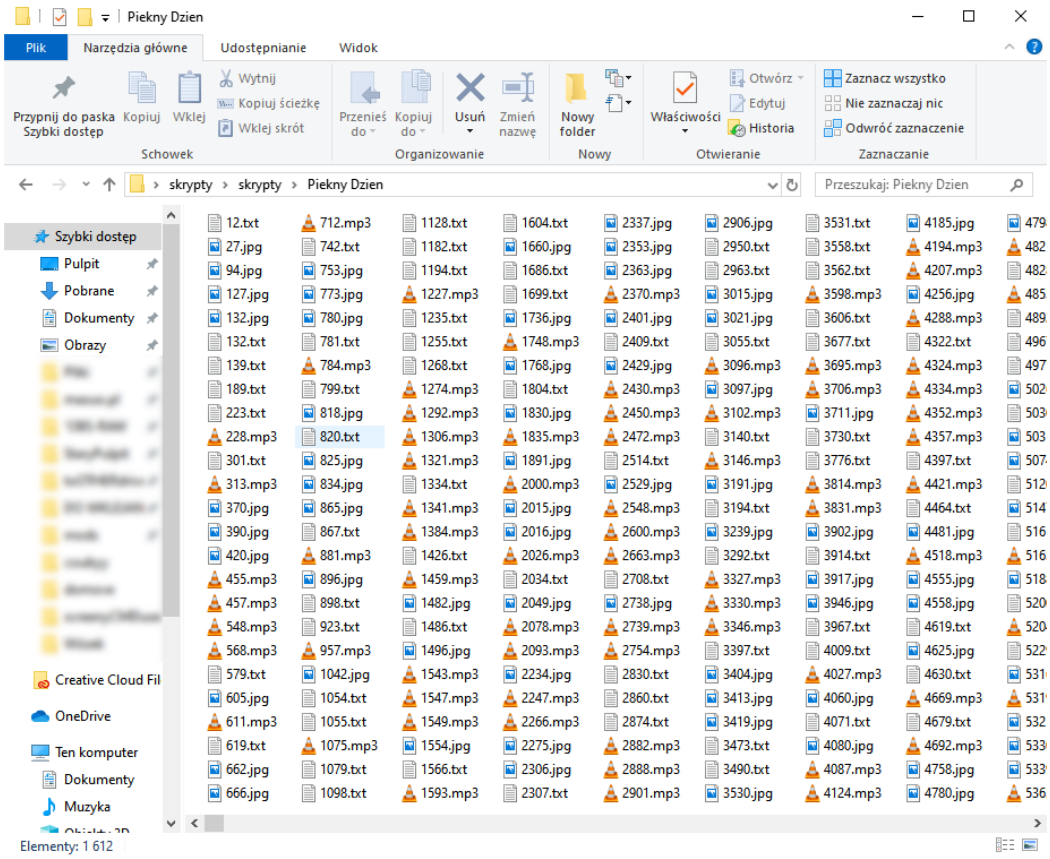
Jest to jeden z przydatniejszych skryptów z tych przykładów. Może nam posłużyć, jeśli chcemy posortować pliki według rozszerzenia, części nazwy, czy innych parametrów. Za przykład posłuży nam folder pełen śmieci ze skryptu #4 oraz drzewko z #5. Skrypt posortuje pliki znajdujące się w folderze „Piekny Dzien” według rozszerzenia oraz rozmieści je w odpowiednich folderach.

polecenie8.bat

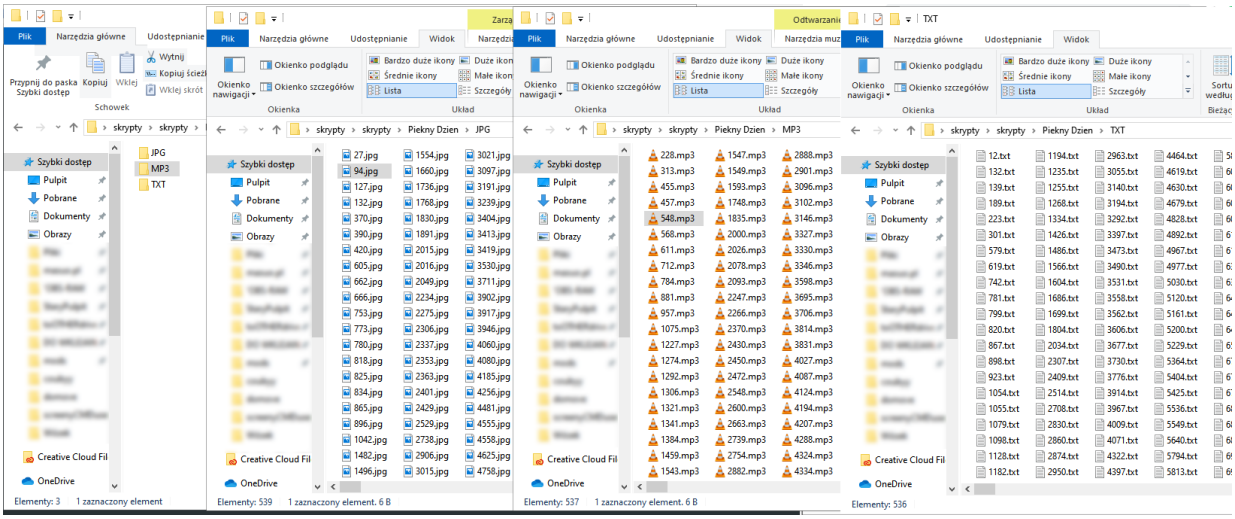
```
@echo off
cd "Piekny Dzien"
mkdir MP3 TXT JPG
echo Utworzono potrzebne katalogi!
move "*.mp3" "MP3"
move "*.txt" "TXT"
move "*.jpg" "JPG"
echo Przeniesiono wszystkie pliki!
pause
```

Obrazek 16: Skrypt #8

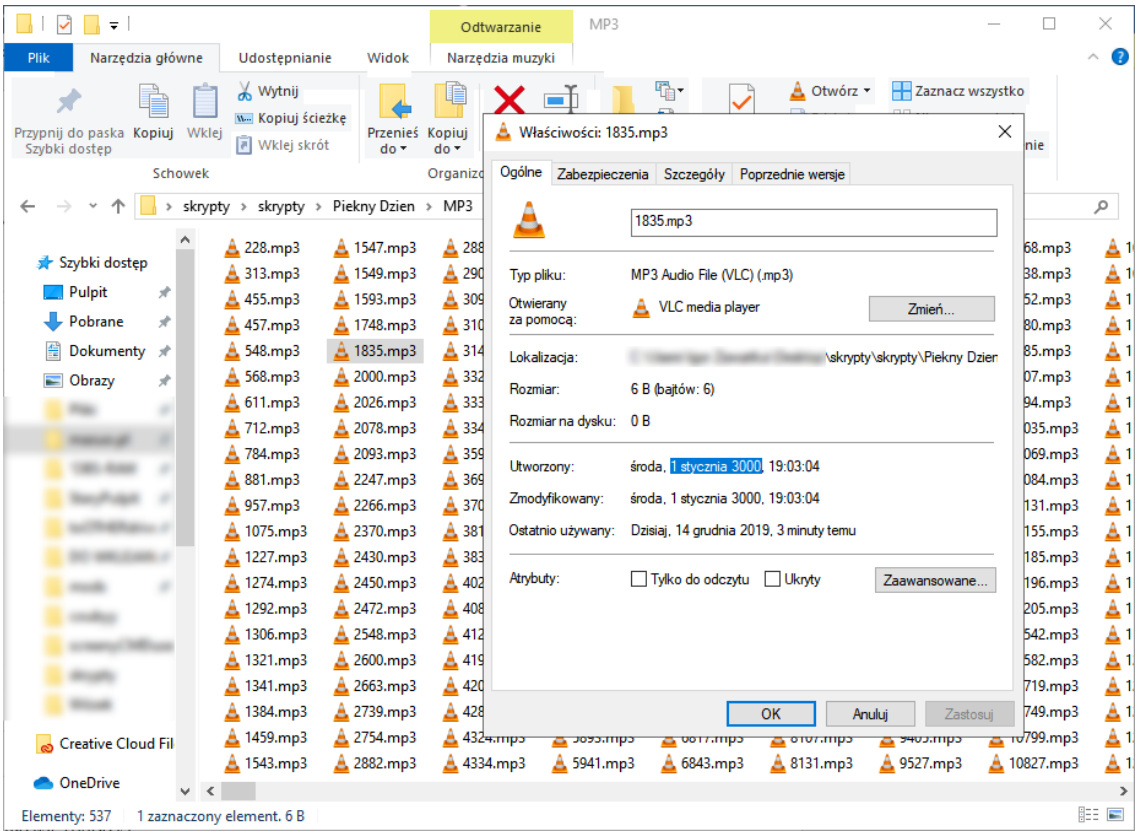
Polecenie move służy do przenoszenia plików. Znak gwiazdki natomiast zastępuje nam dowolny ciąg znaków. Dla przykładu użycie „*.txt” sprawi, że wszystkie pliki, które w swojej nazwie mają „.txt” na końcu, zostaną przeniesione. Gdybyśmy napisali „ab*.txt”, przeniesione zostałyby tylko pliki zaczynające się na „ab” i kończące na „.txt”.



Obrazek 17: Bałagan przed posortowaniem



Obrazek 18: Pliki posortowane do oddzielnych katalogów



Obrazek 19: Zachowanie daty utworzenia

Jak widać, przy przeniesieniu plików tym sposobem, ich data utworzenia i modyfikacji została na roku 3000 pomimo powrotu do 2019.

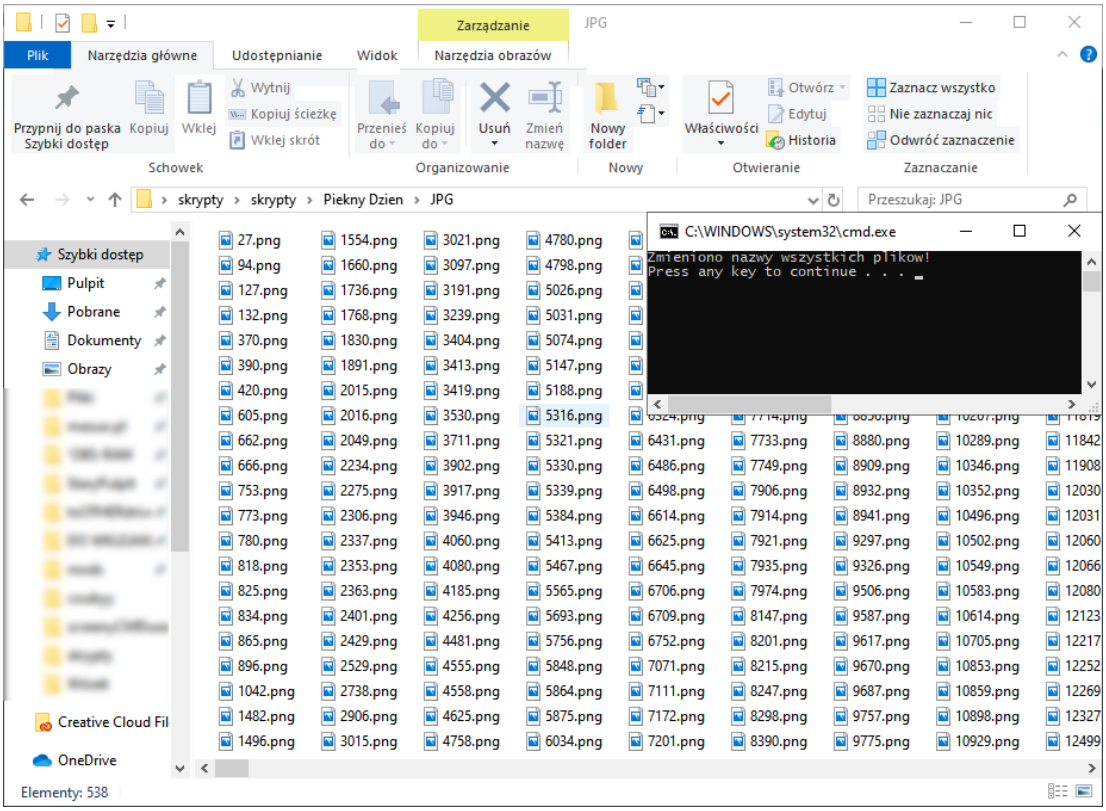
Skrypt #9 – masowa zmiana nazw plików

Jak można się domyśleć, do tego również posłuży magiczna gwiazdka. Do zmiany nazwy pliku użyjemy natomiast polecenia rename. Dla przykładu zmienimy rozszerzenia wszystkich plików .jpg w folderze JPG na .png.

polecenie10.bat

```
@echo off
cd "Piekny Dzien\JPG"
rename "*.jpg" "*.png"
echo Zmieniono nazwy wszystkich plikow!
pause
```

Obrazek 20: Skrypt #9



Obrazek 21: Działanie skryptu #9

Skrypt #10 – ukrycie wszystkich plików

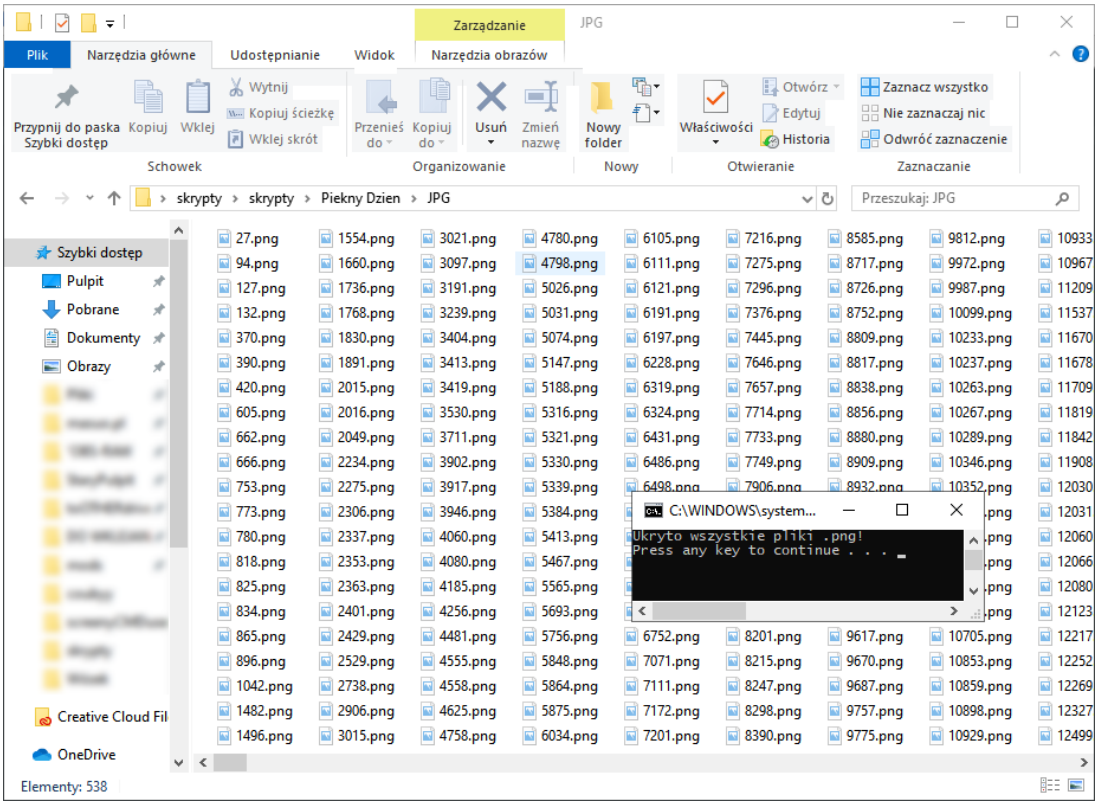
Kolejny skrypt oznaczy wszystkie pliki z folderu JPG jako ukryte. Efekt ten osiągniemy poleceniem attrib +H.

polecenie11.bat

```
@echo off
cd "Piekny Dzien\JPG"
attrib +H
echo Ukryto wszystkie pliki .png!
pause
```

Obrazek 22: Skrypt #10

Polecenie to ma też inne zastosowania, wykonując je bez żadnego argumentu, wypluje nam listę wszystkich plików i katalogów w danym folderze wraz z ich atrybutami, argument -H usunie nam oznaczenie ukrycia, argument +R oznaczy plik jako „tylko do odczytu”, +S jako systemowy itd.



Obrazek 23: Działanie skryptu #10

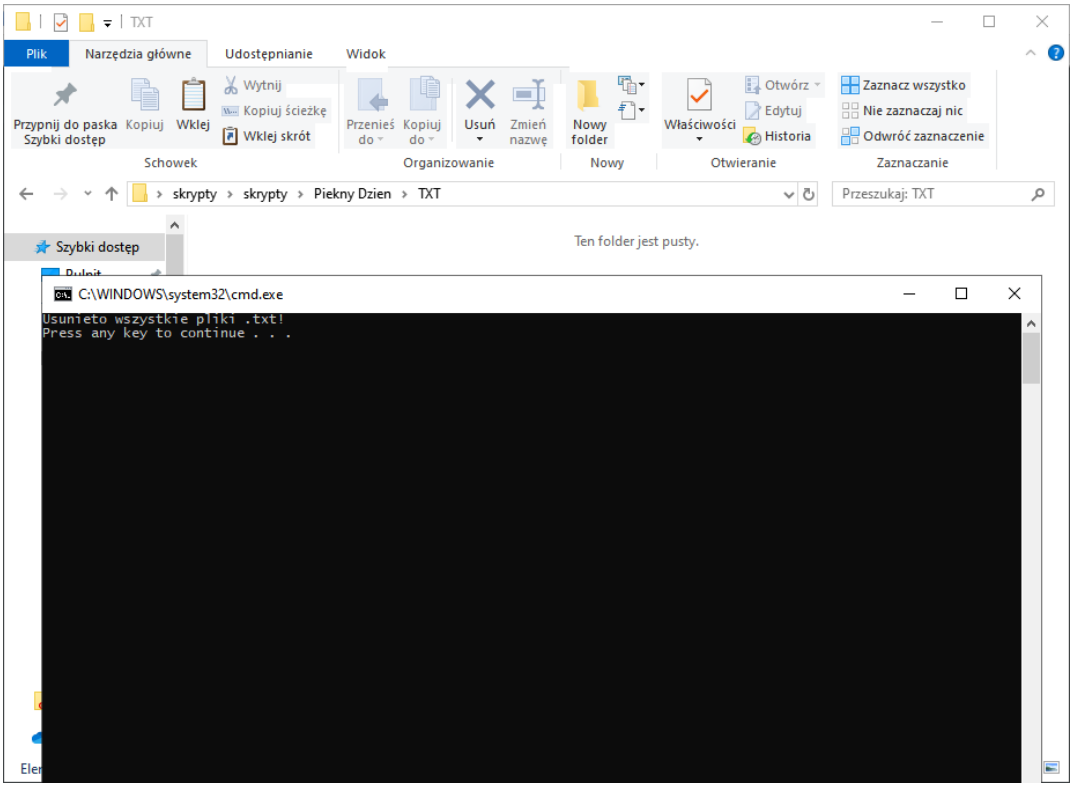
Skrypt #11 – usuwanie plików

W tym przykładzie usuniemy wszystkie pliki z rozszerzeniem .txt. Zrobimy to prostym poleceniem del [nazwa pliku] i po raz kolejny wykorzystamy gwiazdkę.

polecenie12.bat

```
@echo off
cd "Piekny Dzien"
del TXT\*.txt
echo Usunieto wszystkie pliki .txt!
pause
```

Obrazek 24: Skrypt #11



Obrazek 25: Działanie skryptu #11

Natomiast do usunięcia wszystkich plików z danego katalogu, użyjemy „*.*” jako nazwy pliku.

polecenie20.bat

```
@echo off
del TMP\*.*
echo Usunieto!
pause
```

Obrazek 26: Usuwanie wszystkich plików z folderu

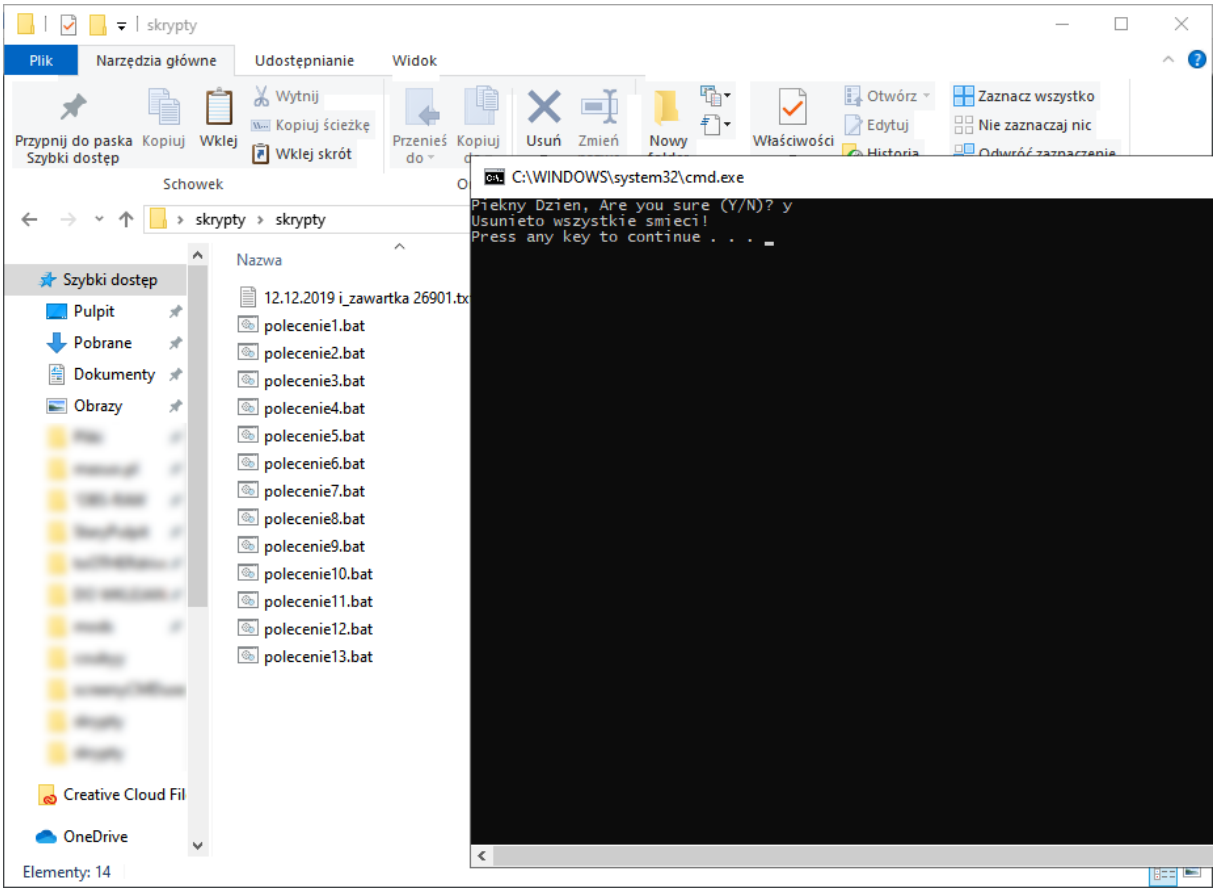
Skrypt #12 – usuwanie folderów

Posłuży nam do tego komenda rd [nazwa katalogu]. Dodatkowo, jeśli nie jest on pusty, musimy dopisać argument /S

polecenie13.bat

```
@echo off
rd "Piekny Dzien" /S
echo Usunieto wszystkie smieci!
pause
```

Obrazek 27: Skrypt #12



Obrazek 28: Działanie skryptu #12

Jak widać, po uruchomieniu programu i zatwierdzeniu wyboru, folder „Piekny Dzien” został usunięty.

Skrypt #13 – pętle

Skrypt ten należy już do nieco bardziej zaawansowanych. Utworzy nam on 60 folderów o losowych nazwach. Aby nie męczyć się z kopiowaniem polecenia rd %random% 60 razy, możemy zastosować tzw. „pętlę”. Kod w pętli będzie wykonywany określoną liczbę razy, a następnie, po rozwiązaniu pętli, interpreter przejdzie po prostu do dalszej części skryptu.

polecenie14.bat

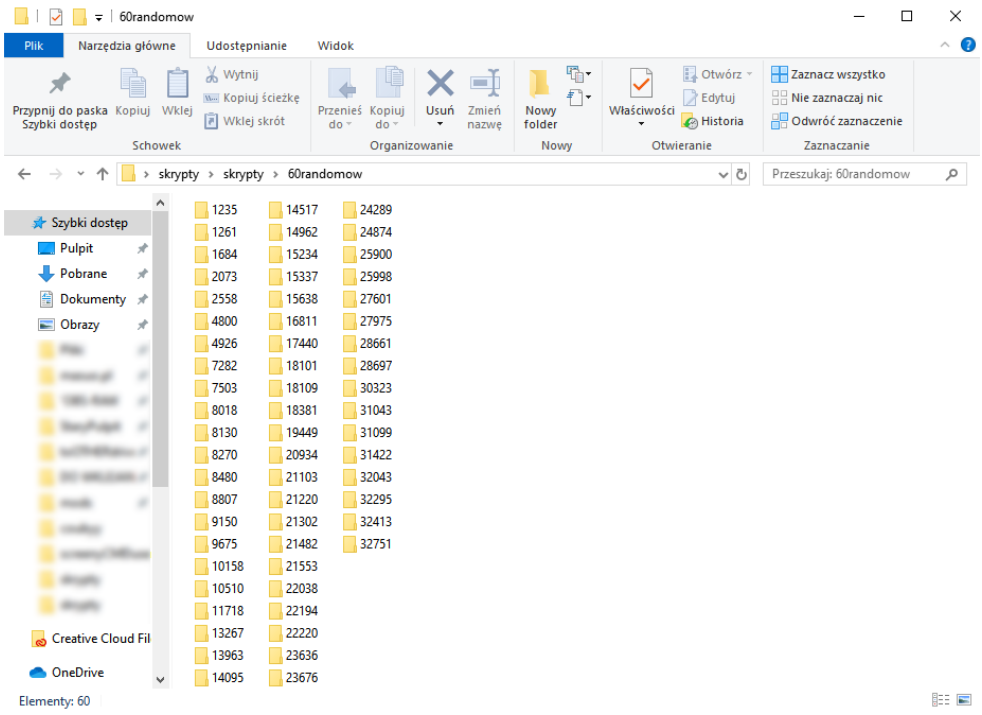
```
@echo off
mkdir 60randomow
cd 60randomow
FOR /L %%A IN (1,1,60) DO (
  mkdir %random%
  echo Utworzono katalog #%%A!
)
echo Utworzono wszystkie 60 katalogów!
pause
```

Obrazek 29: Skrypt #13

Polecenie FOR /L to właśnie pętla. Kod do wykonania w niej zapisujemy w drugiej parze nawiasów, natomiast pełna składnia to:

FOR /L %%[nazwa zmiennej przechowującej liczbę wykonanych iteracji (iterator)] IN ([początkowa wartość iteratora],[wartość, o którą zostanie zwiększony iterator z każym wykonaniem poleceń w pętli],[wartość iteratora, przy której pętla zostanie rozwiązana]) DO ([kod do wykonywania w pętli])

Z początku może wyglądać to nieco zagmatwanie, ale powyższy przykład idealnie to przedstawia. Pętla zacznie się od wartości 1, za każdym „okrążeniem” wartość ta będzie zwiększana o 1, a ostatnim okrążeniem będzie wartość 60.



Obrazek 30: Działanie skryptu #13

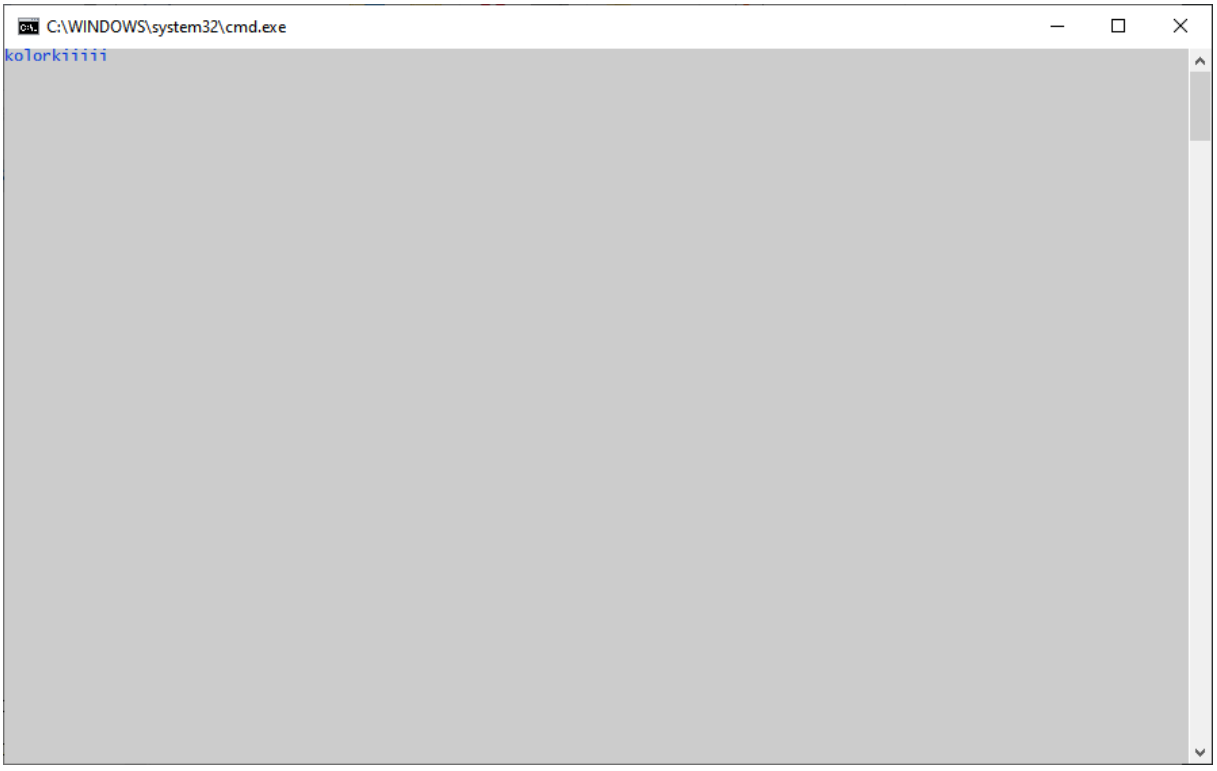
Skrypt #14 – zmiana koloru okna co sekundę

polecenie15.bat

```
@echo off
:START
set /a kolor = %random%%%64+32
color %kolor%
cls
echo kolorkiiiii
TIMEOUT 1 > nul
goto START
```

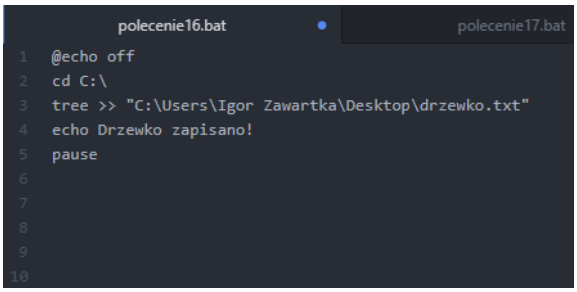
Obrazek 31: Skrypt #14

Pierw definiowana jest zmienna „kolor” i jej wartość to modulo 64 z wartości losowej + 32. Może wydawać się to dziwne, ale w praktyce zwróci nam losową liczbę z przedziału 32 – 64 i zapisze ją do zmiennej kolor. Następnie poleceniem `color %kolor%` kolor konsoli zmienia się na wartość zmiennej (cmd przechowuje kolory właśnie jako liczby do 64, a przedział 32-64 zapewni nam ustawianie tylko jaskrawych kolorów co skutecznie zwiększa szansę na epilepsję u użytkownika komputera ale też ładniej wygląda). Następnie skrypt zostanie zawieszony na sekundę poleceniem `TIMEOUT 1`, a dzięki dopiskowi `> nul`, użytkownikowi nie wyświetli się informacja, że może skrócić ten czas.



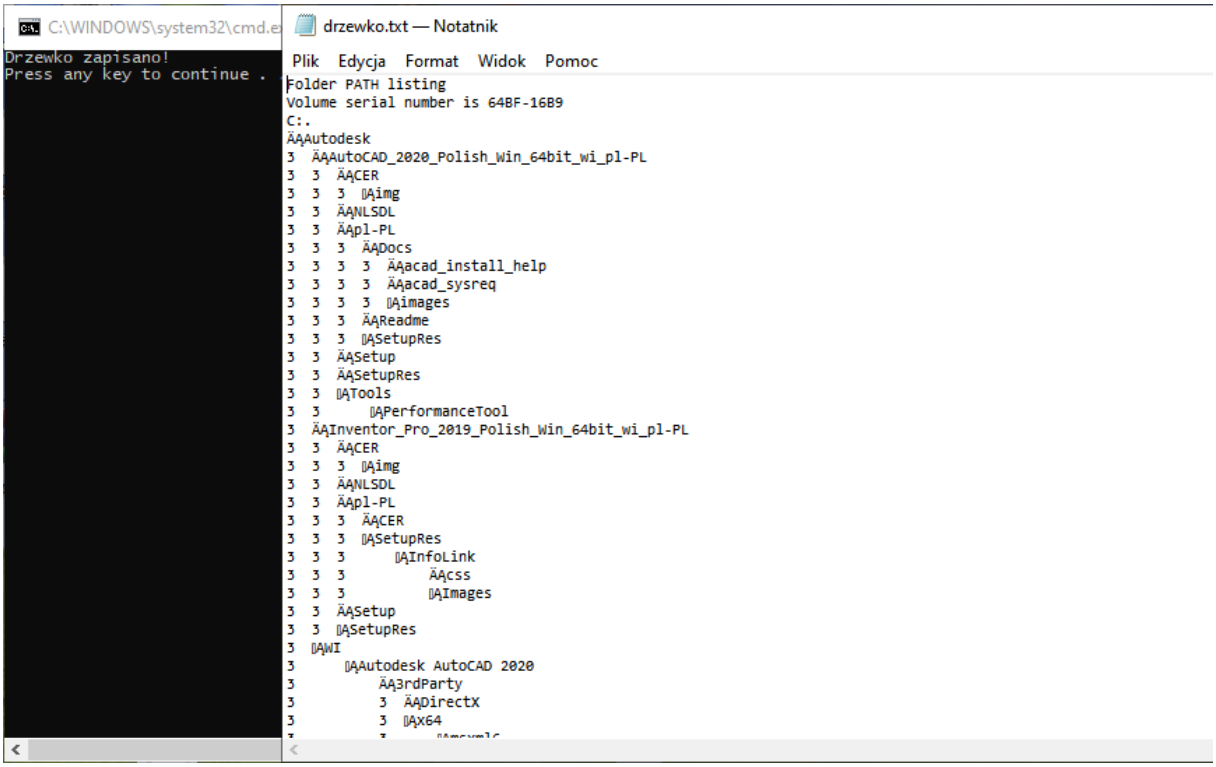
Obrazek 32: Przykładowy kolor wygenerowany przez skrypt #14

Skrypt #15 – zapisanie drzewka dysku C:\ do pliku



Obrazek 33: Skrypt #15

Zwykłe polecenie tree „wyplułoby” drzewko do konsoli. Dzięki użyciu podwójnego znaku większości, drzewko zapisze się do podanego pliku.



Obrazek 34: Działanie skryptu #15

Niestety notatnik domyślnie używa innego kodowania znaków niż cmd, więc piękne słupki i gałązki zostają zastąpione przez dziwne ciągi znaków i litery. W gruncie rzeczy jednak, wszystkie pliki są widoczne i większość nazw powinna być czytelna.

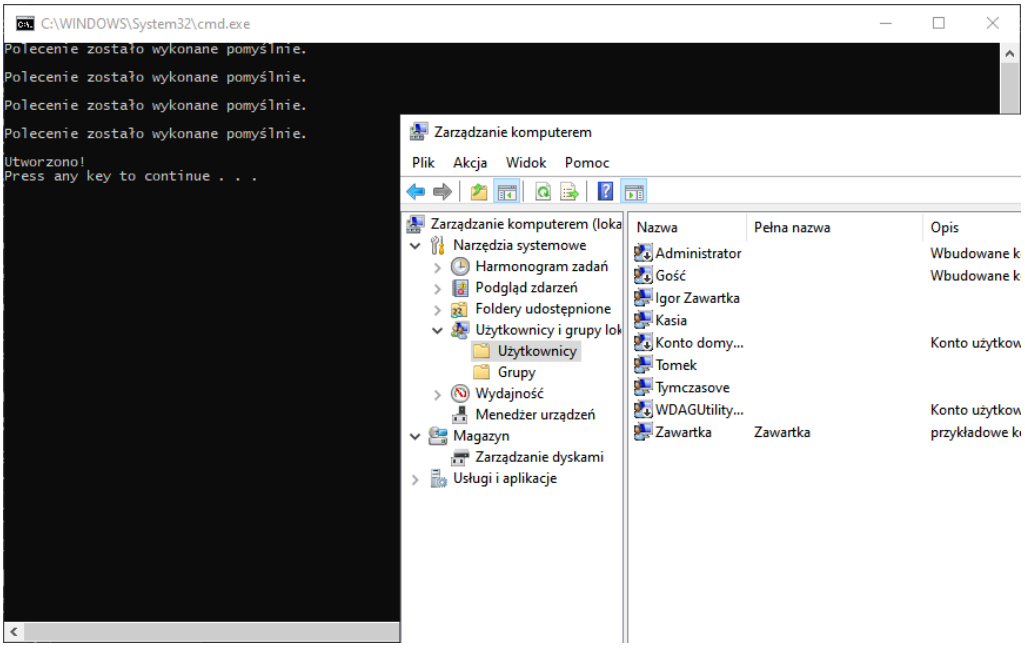
Skrypt #16 – modyfikowanie ustawień kont użytkowników

Skrypty pozwalają nam na wykonywanie wszystkich poleceń, które moglibyśmy wykonać po prostu wpisując je w cmd, a więc możemy także zarządzać przy ich pomocy kontami użytkowników. Dla przykładu napiszmy skrypt, którego zadaniem będzie stworzenie kont Kasia i Tomek, a następnie przydzielenie im czasu logowania od poniedziałku do piątku w godzinach 7:00 – 20:00 oraz w weekendy od 6:00 do 23:00.

polecenie19.bat

```
@echo off
net user Kasia /add
net user Kasia /times:Pn-Pt,7:00-20:00;So-N,6:00-23:00

net user Tomek /add
net user Tomek /times:Pn-Pt,7:00-20:00;So-N,6:00-23:00
echo Utworzono!
pause
```

Obrazek 35: skrypt #16

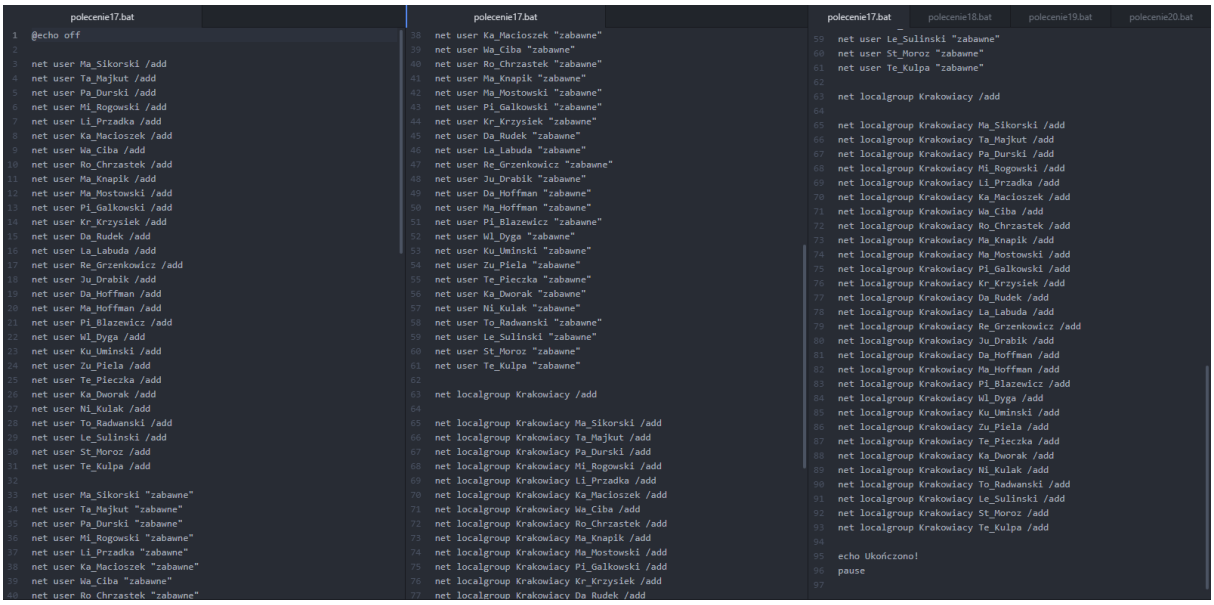


Obrazek 36: Działanie skryptu #16

Jak widać, konta zostały pomyślnie utworzone, a ich godziny logowania ustawione.

Skrypt #17 – masowe operacje na kontach użytkowników

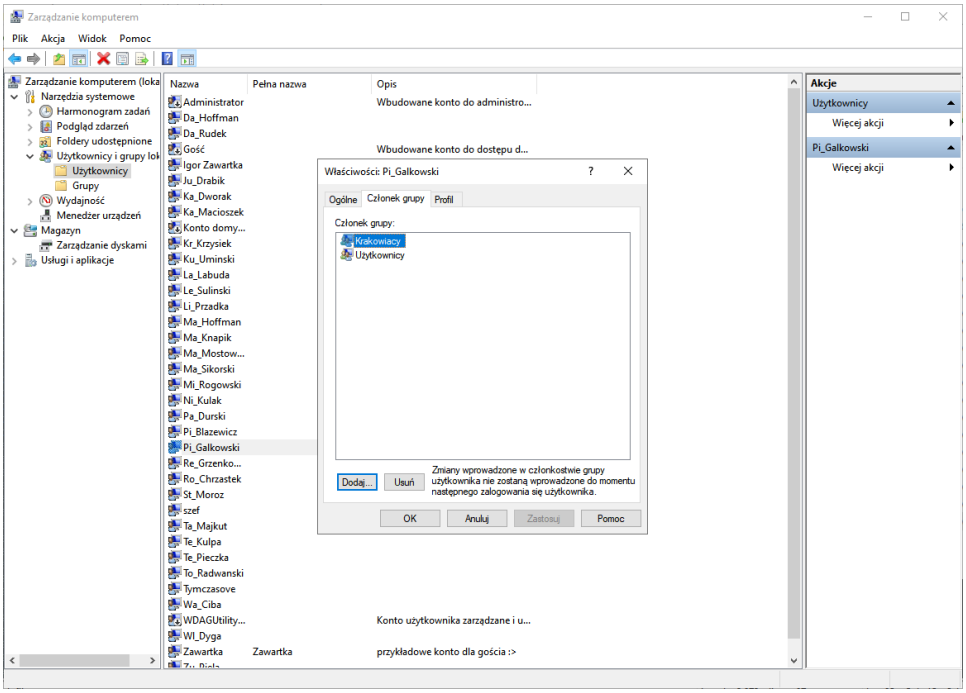
Jest to chyba jedna z ważniejszych zalet skryptów – wykonywanie wielu podobnych operacji jedna po drugiej bez konieczności wpisywania ich wszystkich ręcznie. Im dłuższy skrypt, tym jest to bardziej widoczne. Ten skrypt jest tego idealnym przykładem.



```
@echo off

net user Ma_Sikorski /add
net user Ta_Majkut /add
net user Pa_Durski /add
net user Mi_Rogowski /add
net user Li_Przadka /add
net user Ka_Macioszek /add
net user Wa_Ciba /add
net user Ro_Chrzastek /add
net user Ma_Knapik /add
net user Ma_Mostowski /add
net user Pi_Galkowski /add
net user Kr_Krzysiek /add
net user Da_Rudek /add
net user La_Labuda /add
net user Re_Grzenkowicz /add
net user Ju_Drabik /add
net user Da_Hoffman /add
net user Ma_Hoffman /add
net user Pi_Blazewicz /add
net user Wl_Dyga /add
net user Ku_Uminski /add
net user Zu_Piela /add
net user Te_Pieczka /add
net user Ka_Dworak /add
net user Ni_Kulak /add
net user To_Radwanski /add
net user Le_Sulinski /add
net user St_Moroz /add
net user Te_Kulpa /add

net user Ma_Sikorski "zabawne"
net user Ta_Majkut "zabawne"
net user Pa_Durski "zabawne"
net user Mi_Rogowski "zabawne"
net user Li_Przadka "zabawne"
net user Ka_Macioszek "zabawne"
net user Wa_Ciba "zabawne"
net user Ro_Chrzastek "zabawne"
net user Ma_Knapik "zabawne"
net user Ma_Mostowski "zabawne"
net user Pi_Galkowski "zabawne"
net user Kr_Krzysiek "zabawne"
net user Da_Rudek "zabawne"
net user La_Labuda "zabawne"
net user Re_Grzenkowicz "zabawne"
net user Ju_Drabik "zabawne"
net user Da_Hoffman "zabawne"
net user Ma_Hoffman "zabawne"
net user Pi_Blazewicz "zabawne"
net user Wl_Dyga "zabawne"
net user Ku_Uminski "zabawne"
net user Zu_Piela "zabawne"
net user Te_Pieczka "zabawne"
net user Ka_Dworak "zabawne"
net user Ni_Kulak "zabawne"
net user To_Radwanski "zabawne"
net user Le_Sulinski "zabawne"
net user St_Moroz "zabawne"
net user Te_Kulpa "zabawne"

net localgroup Krakowiacy /add

net localgroup Krakowiacy Ma_Sikorski /add
net localgroup Krakowiacy Ta_Majkut /add
net localgroup Krakowiacy Pa_Durski /add
net localgroup Krakowiacy Mi_Rogowski /add
net localgroup Krakowiacy Li_Przadka /add
net localgroup Krakowiacy Ka_Macioszek /add
net localgroup Krakowiacy Wa_Ciba /add
net localgroup Krakowiacy Ro_Chrzastek /add
net localgroup Krakowiacy Ma_Knapik /add
net localgroup Krakowiacy Ma_Mostowski /add
net localgroup Krakowiacy Pi_Galkowski /add
net localgroup Krakowiacy Kr_Krzysiek /add
net localgroup Krakowiacy Da_Rudek /add
net localgroup Krakowiacy La_Labuda /add
net localgroup Krakowiacy Re_Grzenkowicz /add
net localgroup Krakowiacy Ju_Drabik /add
net localgroup Krakowiacy Da_Hoffman /add
net localgroup Krakowiacy Ma_Hoffman /add
net localgroup Krakowiacy Pi_Blazewicz /add
net localgroup Krakowiacy Wl_Dyga /add
net localgroup Krakowiacy Ku_Uminski /add
net localgroup Krakowiacy Zu_Piela /add
net localgroup Krakowiacy Te_Pieczka /add
net localgroup Krakowiacy Ka_Dworak /add
net localgroup Krakowiacy Ni_Kulak /add
net localgroup Krakowiacy To_Radwanski /add
net localgroup Krakowiacy Le_Sulinski /add
net localgroup Krakowiacy St_Moroz /add
net localgroup Krakowiacy Te_Kulpa /add

echo Ukończono!
pause
```

Obrazek 37: Skrypt #17

Na początku zostaną utworzeni użytkownicy o nazwach [2 pierwsze litery imienia]_[nazwisko]. Następnie skrypt ustawi wszystkim hasła „zabawne” i doda ich do wcześniej utworzonej grupy „Krakowiacy”. Normalnie musielibyśmy wpisywać wszystkie te nazwy i polecenia ręcznie, a tak to nie dość, że możemy sobie tworzenie nazw kont w łatwy sposób zautomatyzować (np. programem Excel), to jeszcze wszystkie polecenia wykonają się automatycznie bez naszej ingerencji, a my możemy tylko patrzeć, czy nie wystąpiły żadne błędy.



Obrazek 38: Działanie skryptu #17

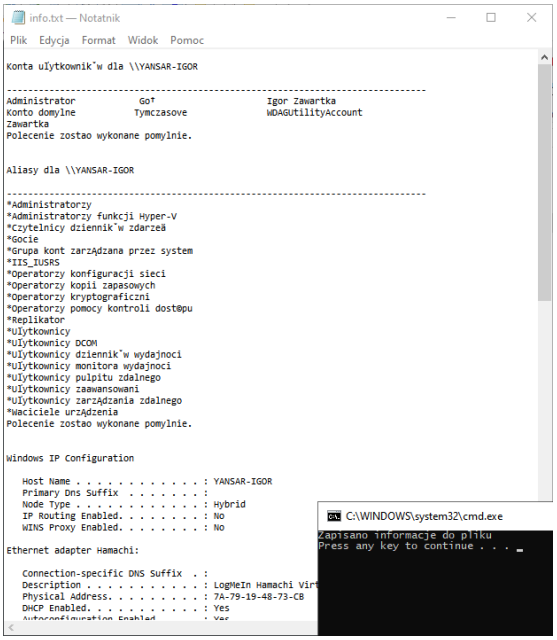
Skrypt #18 – zapisywanie „wypluć” do pliku i autostart

Tak, jak zapisywaliśmy drzewko, możemy także zapisać dowolne inne polecenie. Dla przykładu stworzymy skrypt, który będzie wrzucał do pliku „info.txt” listę wszystkich kont użytkowników, grup oraz informacje o kartach sieciowych komputera. Ponadto chcemy, aby program był wykonywany za każdym razem, gdy uruchomimy komputer.

polecenie18.bat

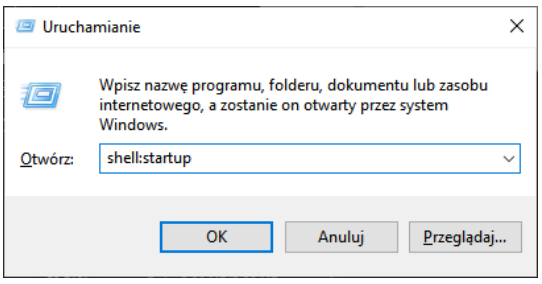
```
@echo off
net user > info.txt
net localgroup >> info.txt
ipconfig /all >> info.txt
echo Zapisano informacje do pliku
pause
```

Obrazek 39: Skrypt #18



Obrazek 40: Działanie skryptu #18

Skrypt jest dosyć prosty i nie wnosi za wiele. Aby zaś program był uruchamiany za każdym razem, gdy włączamy komputer, musimy dodać go do folderu „autostart”. Folder ukaże nam się po wpisaniu w okno uruchamiania „shell:startup” i zatwierdzeniu enterem. Teraz wystarczy już tylko wrzucić do tego folderu wcześniej utworzony skrypt.



Obrazek 41: Otwieranie folderu autostartu