

https://www.tutorialspoint.com/perl/perl\_syntax.htm

*print ”Hello world\n”;*

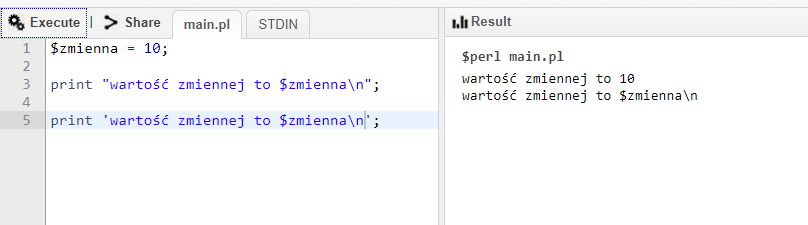
n jest po to żeby było przejście do kolejnej linijki

średnik jest na końcu każdego polecenia

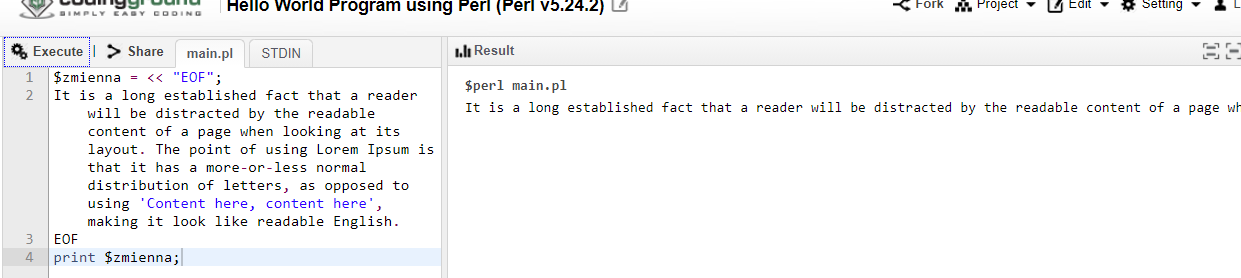
PERL – practical extraction and report language

Larry Wall

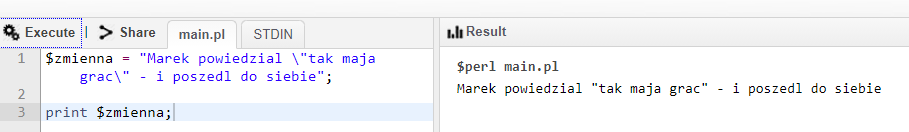
Jeśli w print użyjemy pojedynczego nawiasu zamiast podwójnego to zmienna będzie traktowana jako zwykły tekst:



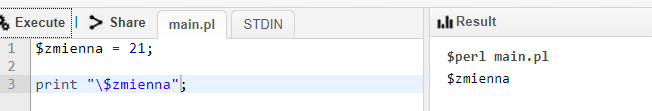
EOF – do przechowywania długich stringów



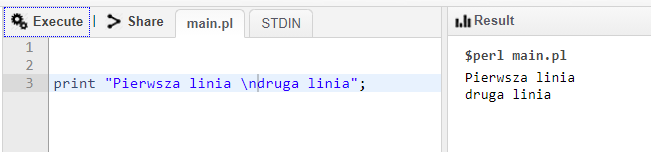
Żeby uwzględnić cudzysłów w stringu należy go poprzedzić znakiem \



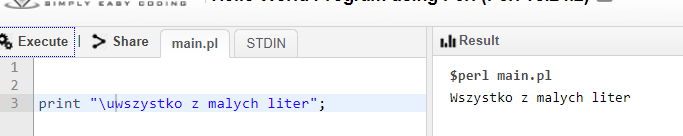
Jeśli użyjemy znaku \ przed $ to print potraktuje to jako tekst a nie jako zmienną:



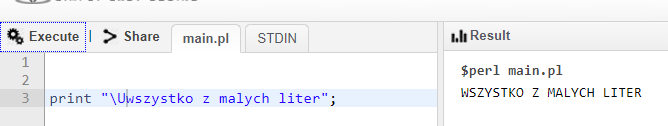
Przykład interpolacji czyli przejścia do nowej linii:



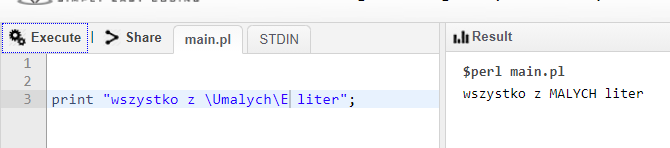
Wielka litera:



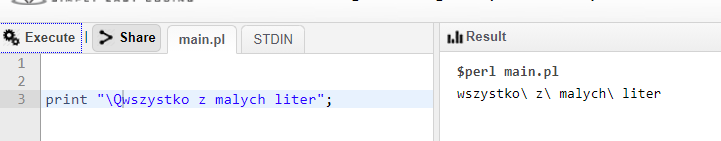
Wielka litera dla całego tekstu:



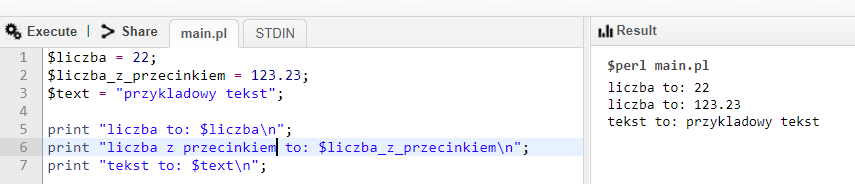
Wielka litera dla fragmentu tekstu – od \U do \E:



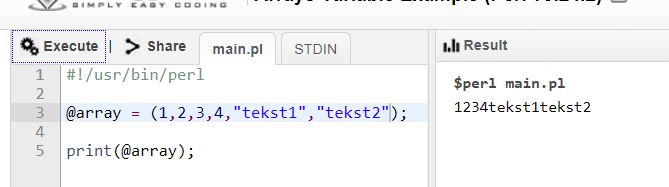
Znak \ przed każdym znakiem niealfanumerycznym (czyli np. spacją):



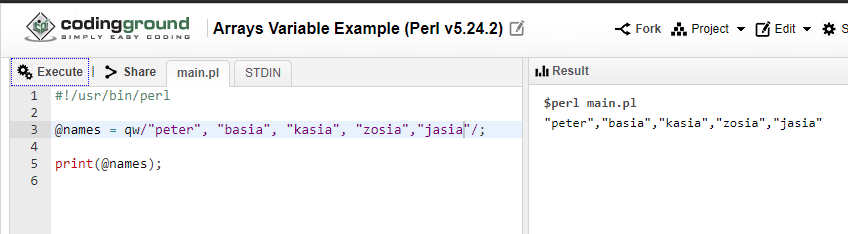
Typy danych – skalar. Może być liczbą albo tekstem:



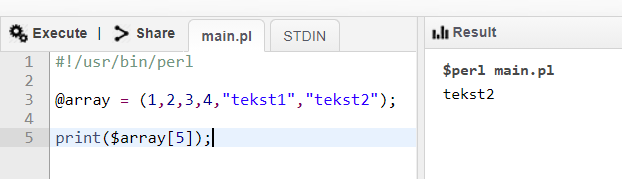
Typ danych – array czyli macierz (chyba):

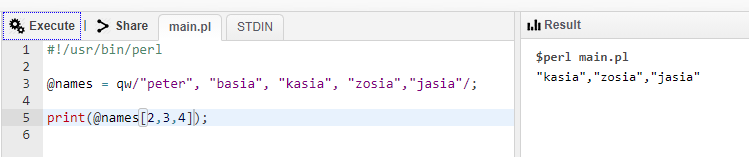


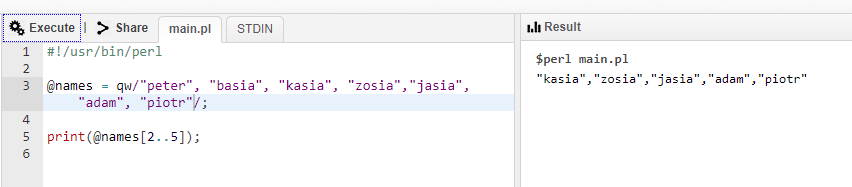
Inny sposób na deklarację macierzy:



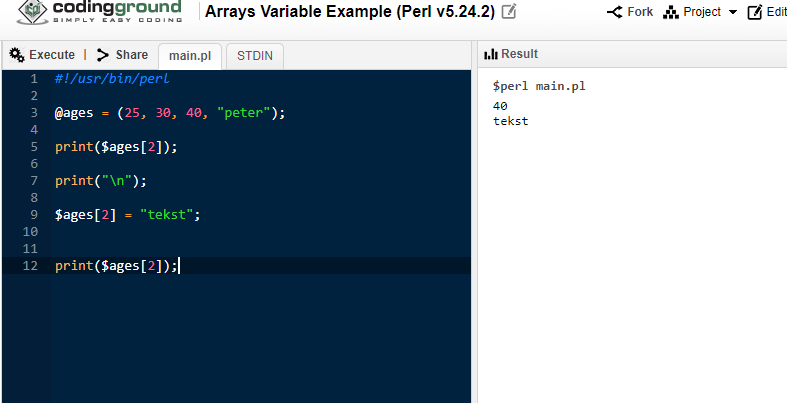
W taki sposób odwołujemy się do poszczególnych elementów macierzy:



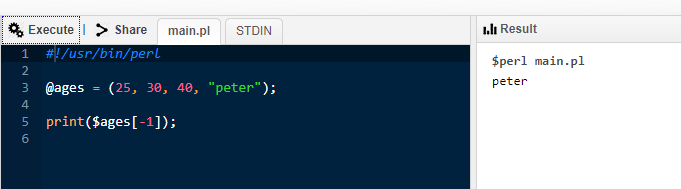




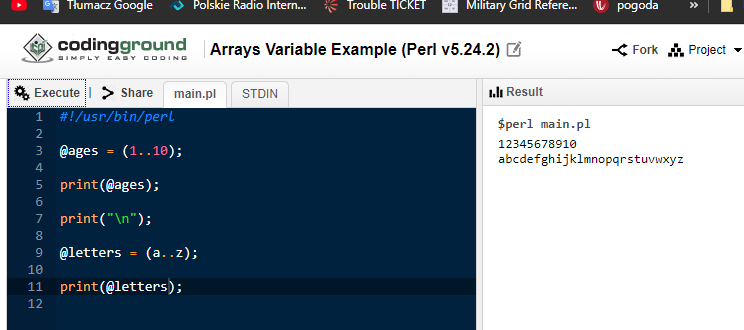
Wstawianie wartości do zmiennych w macierzy:



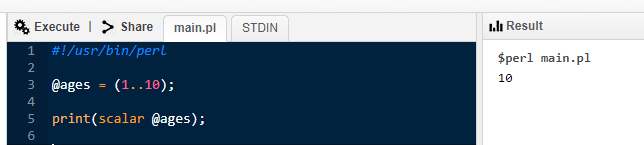
Można odwoływać się do elementów macierzy od tyłu:



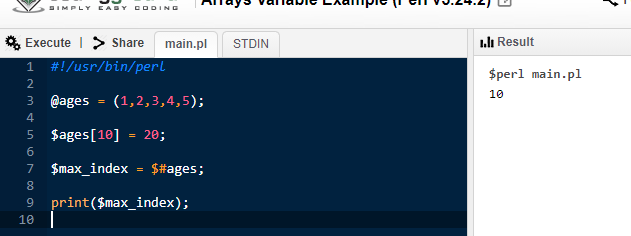
Sekwencyjne tworzenie macierzy:



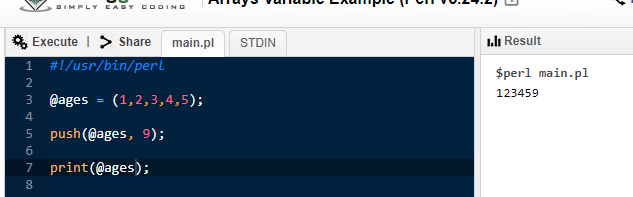
Sprawdzanie rozmiaru macierzy:



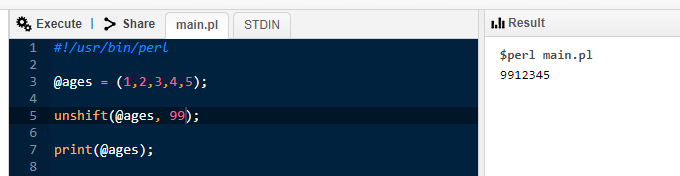
Macierz nie musi być „ciągła”. Możemy sprawdzić maksymalny indeks macierzy:



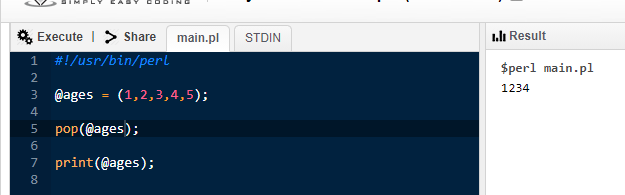
Dodawanie elementów na końcu macierzy:



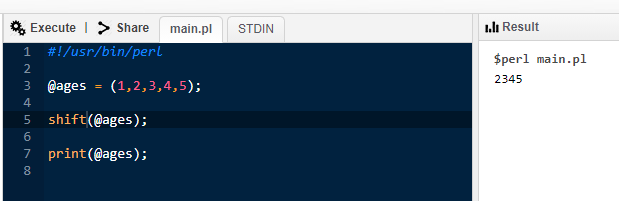
Dodawanie elementu na początku macierzy:



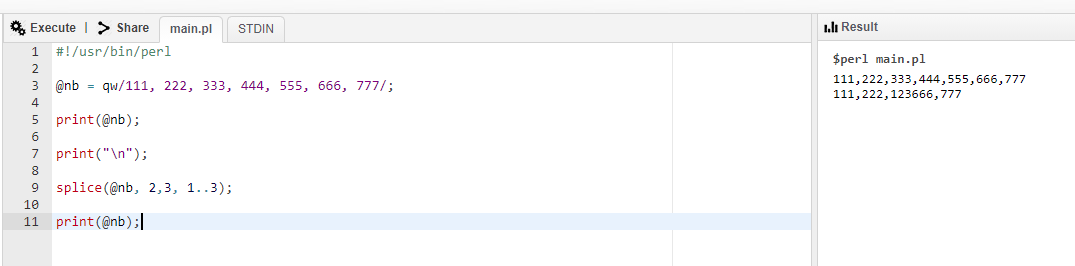
Usuwanie ostatniego elementu listy:



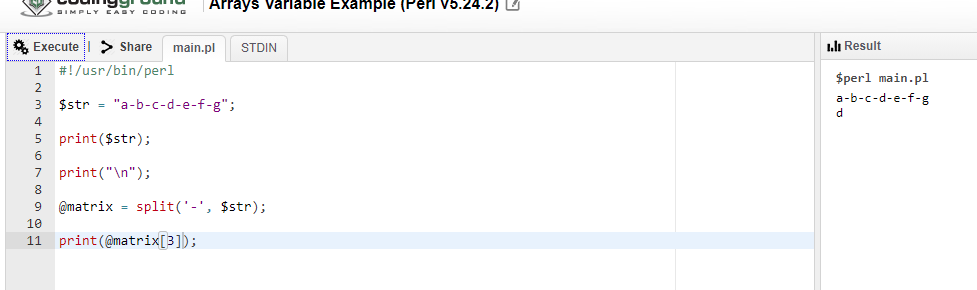
Usuwanie pierwszego elementu listy:



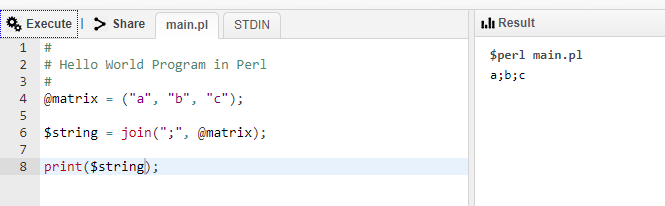
Zastępowanie wybranych elementów macierzy:



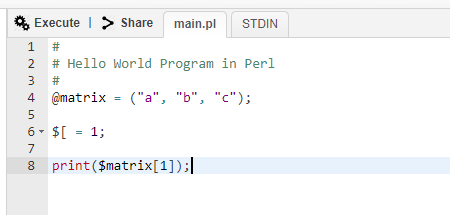
Zmiana stringa w macierz:



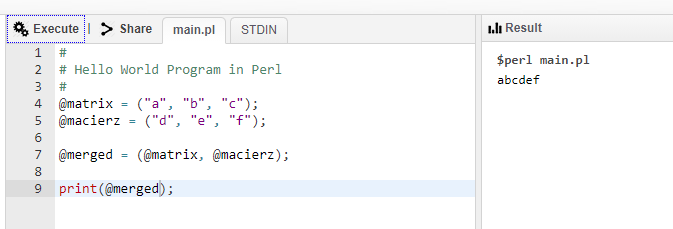
Zmiana macierzy w string:



Macierze są numerowane od 0 ale można zmienić to na inną wartość:

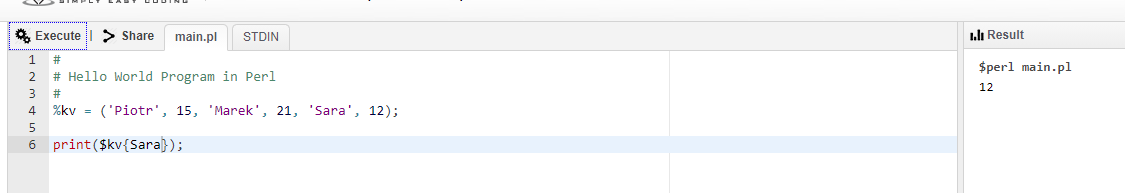


Łączenie macierzy:

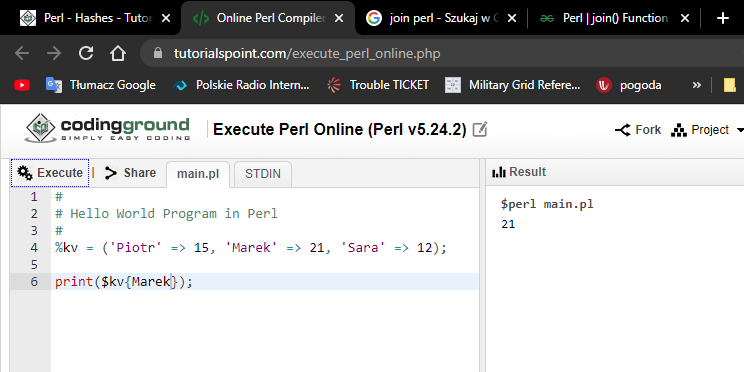


Typ danych klucz-wartość:

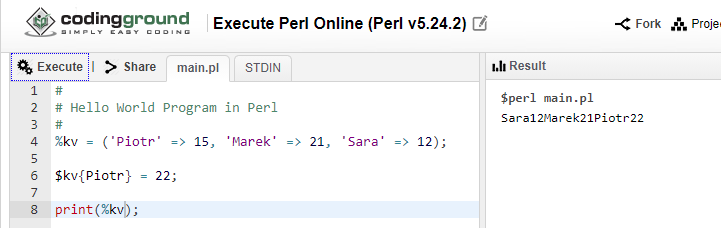
Deklaracja i odwołanie do wartości:



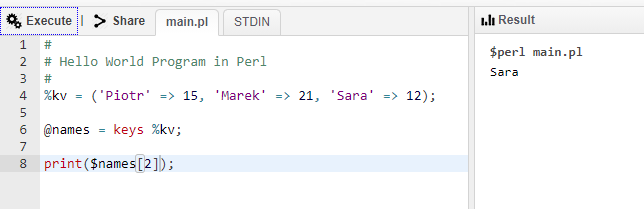
Można też deklarować w taki sposób żeby nie było wątpliwości co jest kluczem a co wartością:

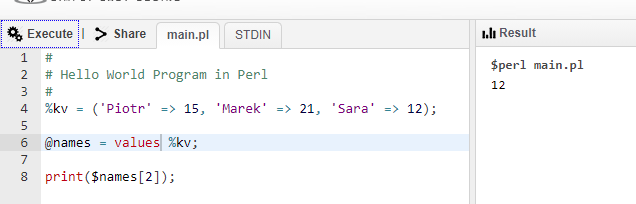


Zmiana wartości:

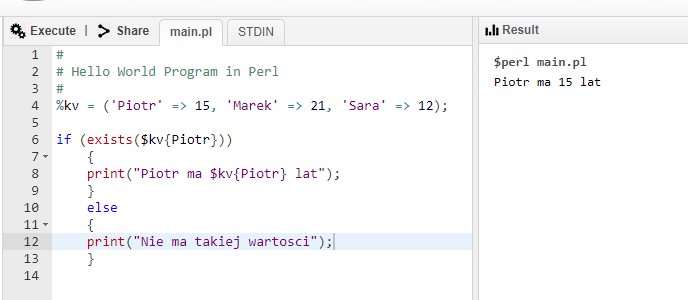


Odwołanie do kv jak do macierzy wywyołując klucz lub wartość:

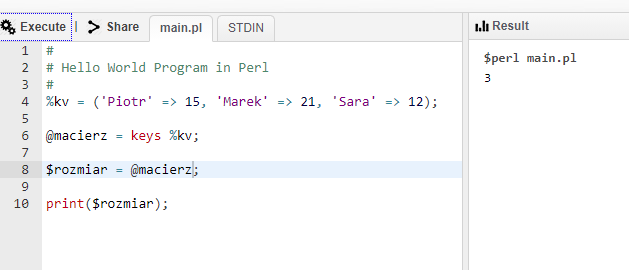




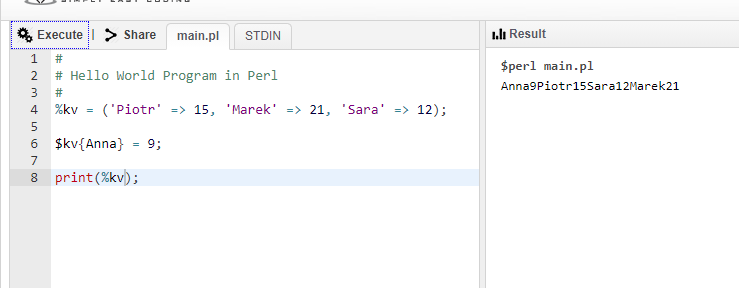
Sprawdzanie czy podany klucz istnieje i funkcja if:



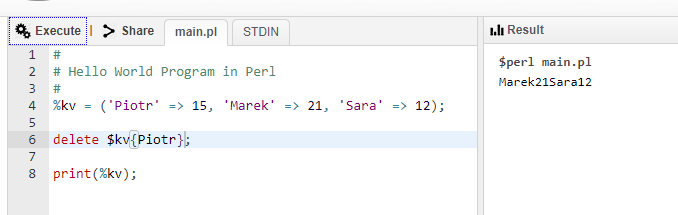
Sprawdzane rozmiaru kv:



Dodawanie kolejnej wartości do kv:

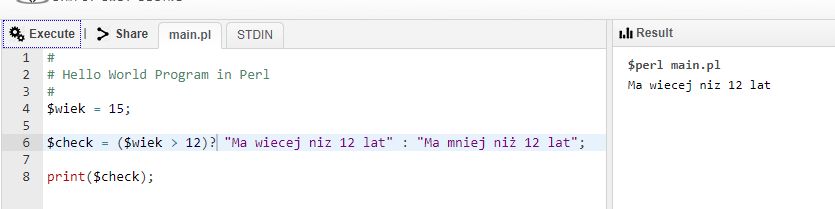


Usuwanie wartości z kv:

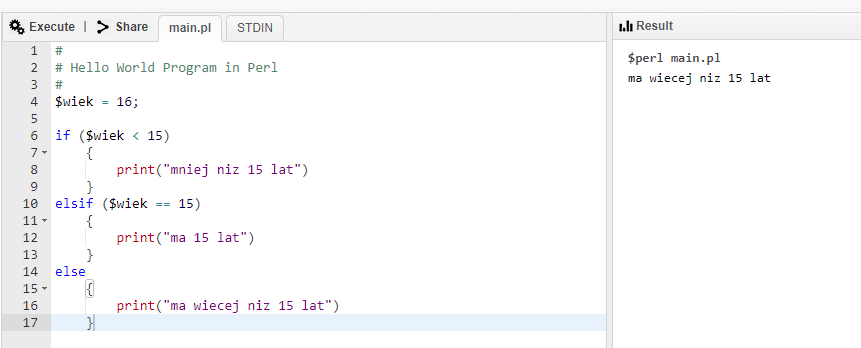


If than else:

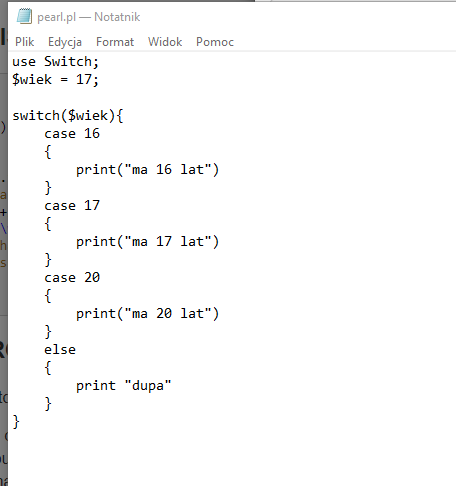
Najprostrza forma:

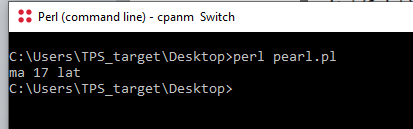


Zwykła forma:

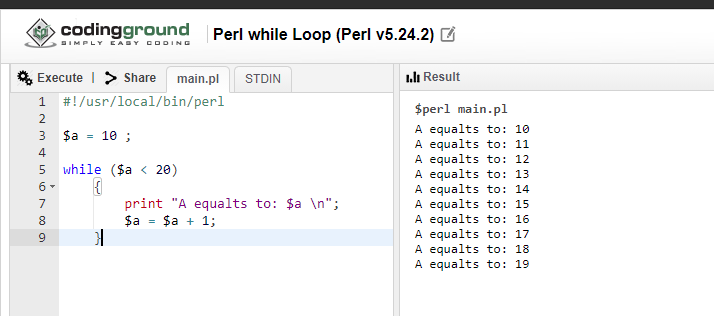


Switch:

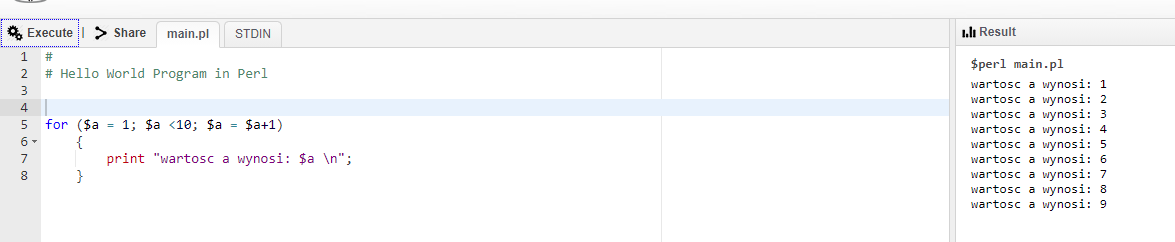




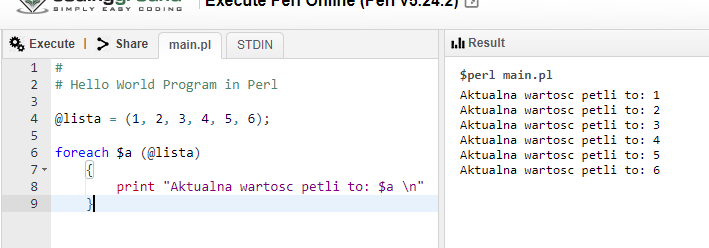
Pętla while:



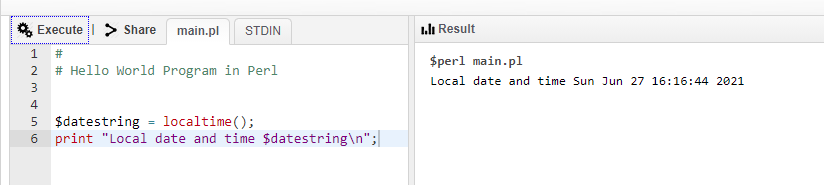
Pętla for:

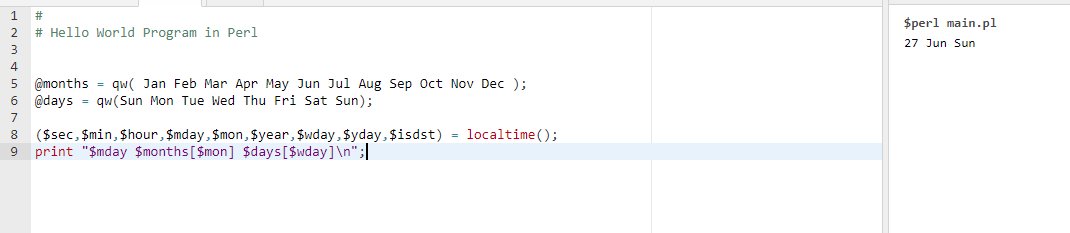


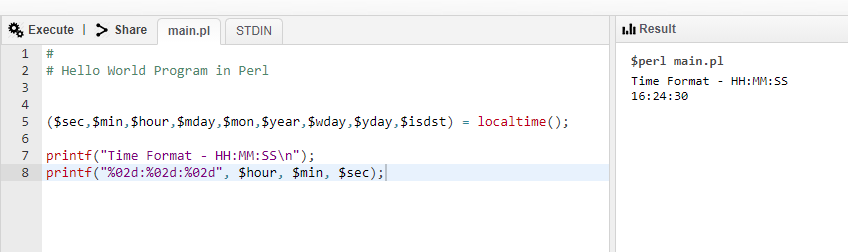
Pętla foreach:



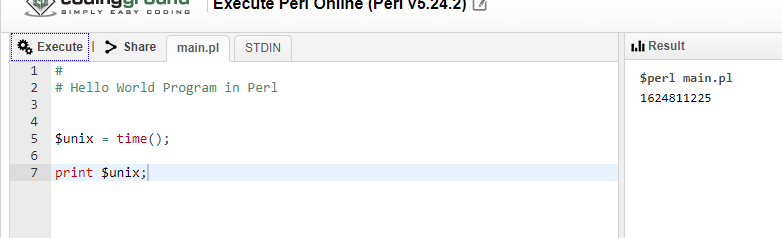
Data:



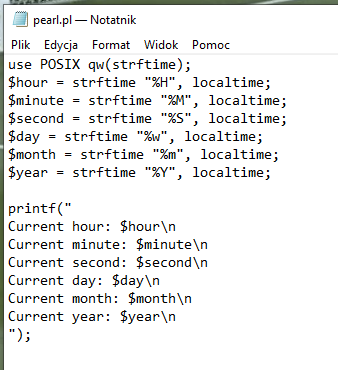


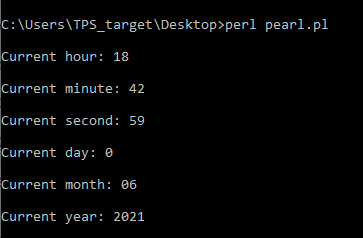


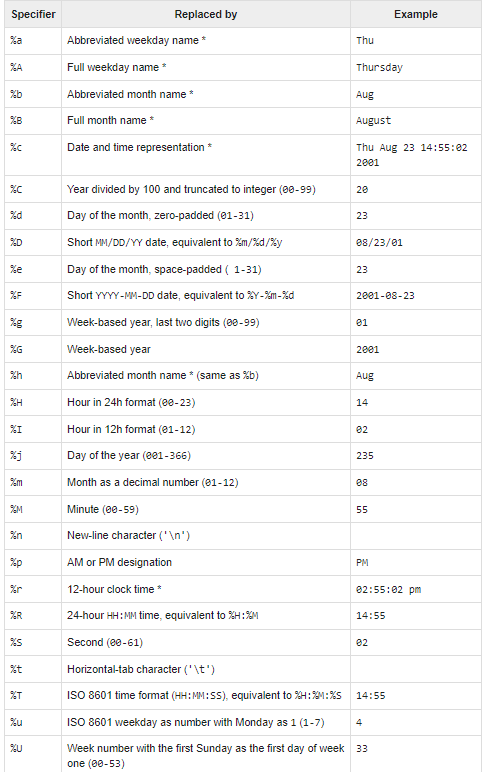
Czas unixowy:

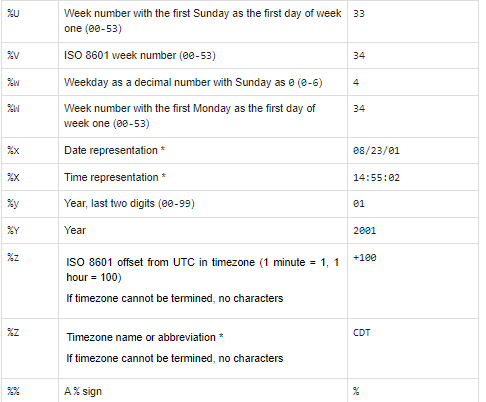


Maski daty:



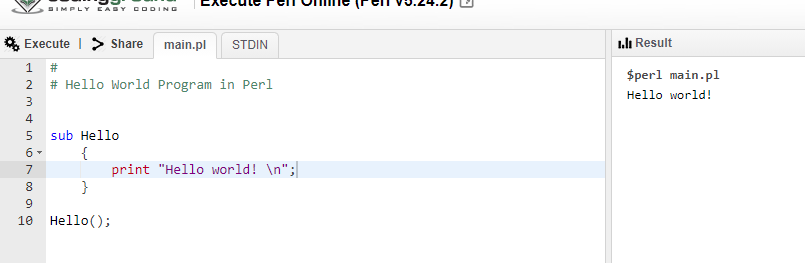




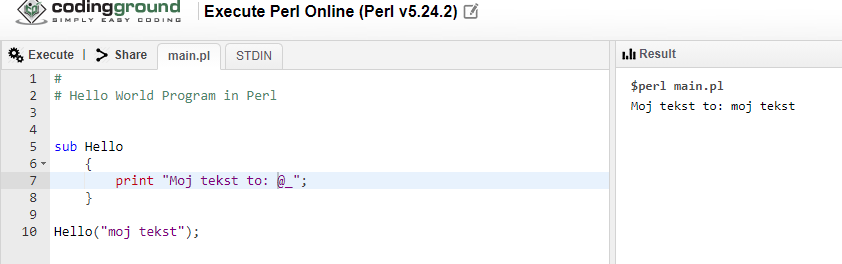


Subroutines

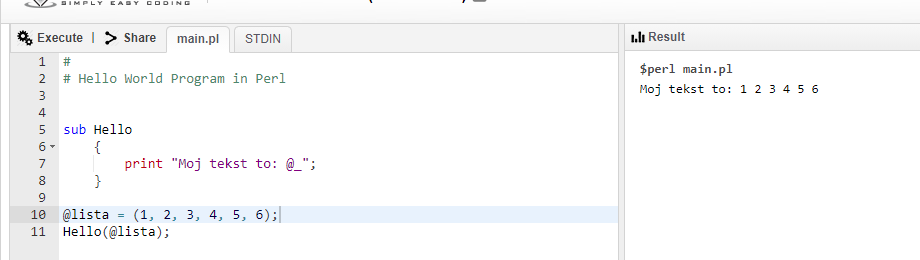
Coś jak funkcja:



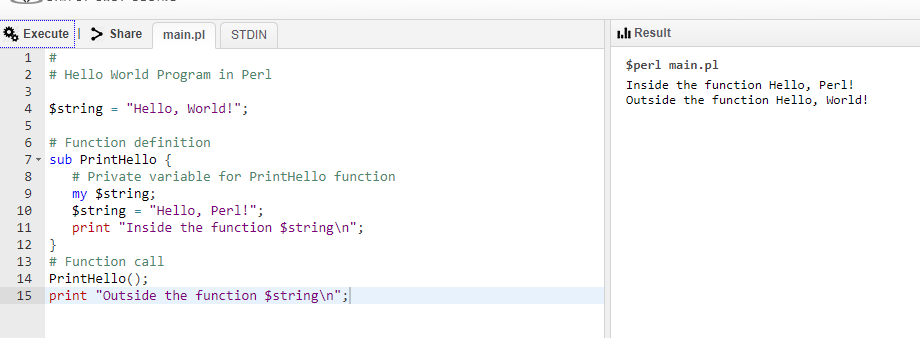
Funkcja z parametrem:



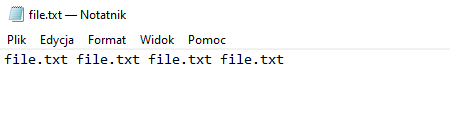
Jako argument można podawać listę:

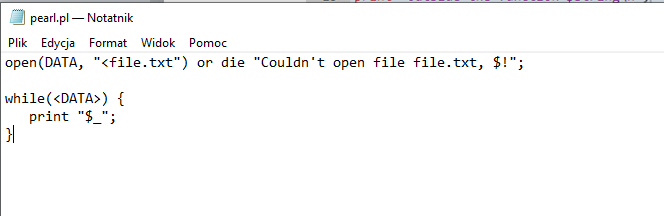


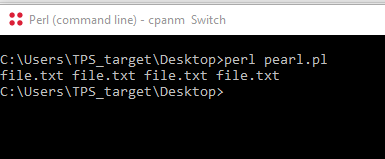
Zmienne prywatne i globalne:



Odczyt z pliku txt umiezczonego w tej samej lokalizacji co odpalany:



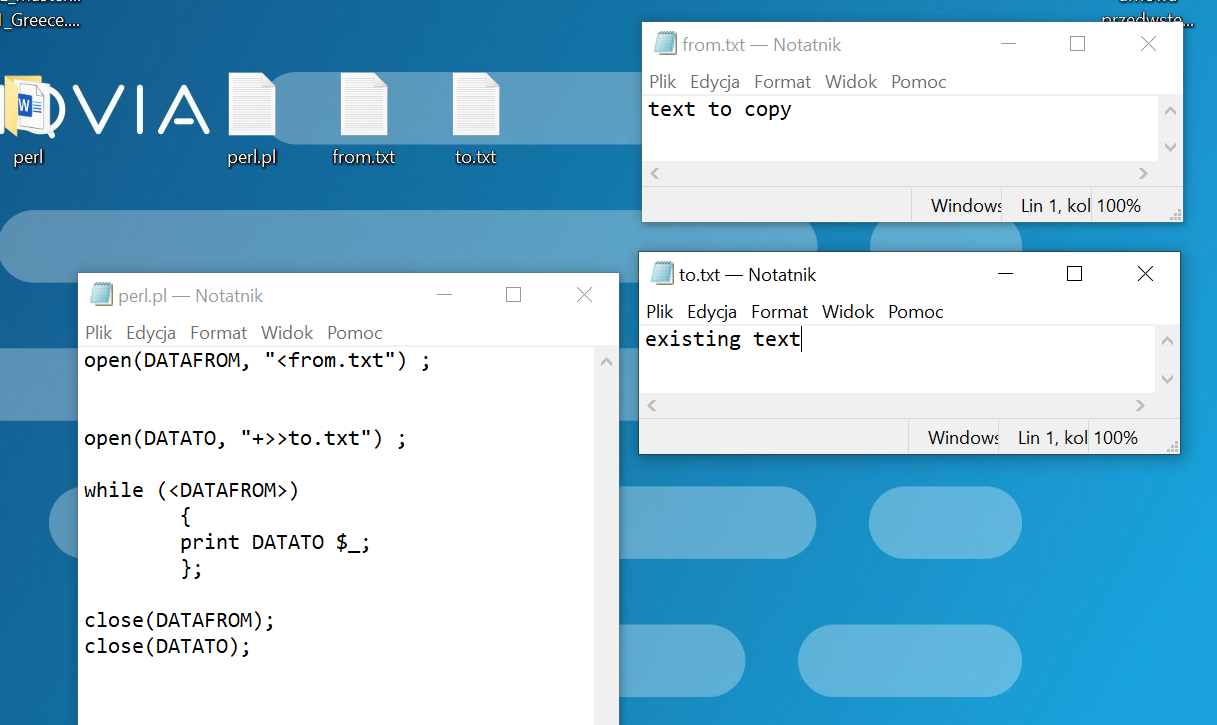




open(DATA, "<file.txt")

Zaznaczony fragment odpowiada za to, że odpalamy plik w trybie read only.

Kopiowanie z pliku do pliku:



Zawartość pliku po odpaleniu skryptu:

