# ALGORYTMY

## 1. Temat: ALGORYTMICZNE ROZWIĄZYWANIE PROBLEMÓW – POWTÓRZENIE I UZUPEŁNIENIE

Notatka:

Programowanie (tworzenie programu) rozpoczyna się od ułożenia **algorytmu**, według którego będzie działał program, potem ten algorytm zapisuje się w języku zrozumiałym dla komputera.
**Algorytm** – to opis czynności, które należy wykonać w ściśle określonym porządku, aby rozwiązać dany problem (grupy zadań).
**Program komputerowy** – to ciąg instrukcji przeznaczonych dla komputera, zapisanych w języku programowania.
**Program źródłowy** – to program napisany w języku programowania (np. Logo, Turbo Pascal, C++, itp.), nie może być bezpośrednio wykonany przez komputer, tłumaczony na język wewnętrzny.
**Język wewnętrzny komputera –** to ciąg zrozumiałych dla komputera operacji elementarnych, zw. **programem (kodem) wynikowym**.
**Translacja –** to tłumaczenie programu źródłowego na program wynikowy.
**Translator –** to program tłumaczący. Wyróżnia się ich dwa rodzaje: **interpretator, kompilator.**
**Program** napisany w języku **Logo** może:

- wykonywać obliczenia
- pisać teksty
- tworzyć rysunki na ekranie

**Język Logo –** to język strukturalny, umożliwiający dzielenie algorytmu na wyraźnie wyodrębnione problemy, których rozwiązanie opisuje się za pomocą **procedur** (tzn. poleceń zrozumiałych dla tego języka). W programie Logo Komeniusz Procedury pierwotne dzielimy na polecenia (**instrukcje**) i **funkcje** (procedury, które mają wartość).

## 2. Temat: ŚRODOWISKO PROGRAMU LOGO KOMENIUSZ – TWORZENIE PROSTYCH RYSUNKÓW

Notatka:

Procedury pierwotne (polecenia): (po wpisaniu procedury wciskamy klawisz Enter)
**pż –** (pokaż żółwia) żółw pokazuje się na ekranie
**sż –** (schowaj żółwia) żółw przestaje być widoczny na ekranie
**naprzód** ***liczba_kroków*** lub **np** ***liczba_kroków*****–** przesunięcie żółwia do przodu o podaną liczbę kroków (np. np 100)
**wstecz** ***liczba_kroków*** lub **ws** ***liczba_kroków*** – przesunięcie żółwia do tyłu o podaną liczbę kroków (np. ws 70)
**prawo** ***kąt*** lub **pw** ***kąt*** – obrót żółwia w prawo o podany w stopniach kąt (np. pw 90)
**lewo** ***kąt*** lub **lw** ***kąt*** – obrót żółwia w lewo o podany w stopniach kąt (np. lw 45)
**podnieś** lub **pod –** powoduje podniesienie pisaka żółwia
**opuść** lub **opu -** powoduje opuszczenie pisaka żółwia
**zmaż –** wymazuje wszystkie rysunki z ekranu bez zmiany pozycji żółwia
**wróć –** ustawia żółwia w pozycji początkowej na środku ekranu
**czyść** lub **cs** - wymazuje wszystkie rysunki z ekranu i ustawia żółwia w pozycji początkowej
**UstalGrubośćPisaka** *grubość* lub **ugp** *grubość* -ustala jaką szerokość ma mieć „rysik" pisaka
**UstalKolorPisaka** *nr_koloru* lub **ukp** *nr_koloru* - ustala kolor pisaka, pod numerami od 1 do 15 są poszczególne kolory
**UstalTło** *nr_koloru* - ustala kolor tła
**zamaluj -** Należy podnieść pióro, ustawić się wewnątrz figury zamkniętej i opuścić pióro. Zamaluje (wypełni) wnętrze tej figury kolorem, który jest aktualnie ustalony (poleceniem ukp)
**powtórz** ***n*** **[*lista_poleceń*]** - powoduje n-krotne powtórzenie listy poleceń (np. powtórz 2 [np. 50 pw 90])

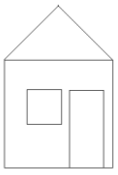
Kartki z ćwiczeniami do lekcji: **Procedury pierwotne** (polecenia): (po wpisaniu procedury wciskamy klawisz **Enter**)

| Polecenia | Skrót | Znaczenie | Działanie | Przykład |
|---|---|---|---|---|
| **pż** | | „pokaż żółwia" | żółw pokazuje się na ekranie | pż |
| **sż** | | „schowaj | żółw przestaje być widoczny na ekranie | sż |

| | | | | |
|---|---|---|---|---|
| | | żółwia" | | |
| **naprzód** *liczba_kroków* | **np** *liczba_kroków* | „idź do przodu" | przesunięcie żółwia do przodu o podaną liczbę kroków (np. o 100) | naprzód 100<br>np 100 |
| **czyść** | **cs** | „wyczyść ekran" | wymazuje wszystkie rysunki z ekranu i ustawia żółwia w pozycji początkowej, tzn. na środku | czyść<br>cs |
| **wstecz** *liczba_kroków* | **ws** *liczba_kroków* | „idź wstecz" | przesunięcie żółwia do tyłu o podaną liczbę kroków (np. o 70) | wstecz 70<br>ws 70 |
| **prawo** *kąt* | **pw** *kąt* | „obróć się w prawo o kąt …" | obrót żółwia w prawo o podany w stopniach kąt np. o $90^0$ | prawo 90<br>pw 90 |
| **lewo** *kąt* | **lw** *kąt* | „obróć się w lewo o kąt …" | obrót żółwia w lewo o podany w stopniach kąt np. o $45^0$ | lewo 45<br>lw 45 |
| **podnieś** | **pod** | „podnieś pisaka" | powoduje podniesienie pisaka żółwia, ten nic nie będzie rysował, gdy go będziemy przesuwać | podnieś<br>pod |
| **opuść** | **opu** | „opuść pisaka" | powoduje opuszczenie pisaka żółwia, przy przesuwaniu znów będzie rysował | opuść<br>opu |
| **zmaż** | | | wymazuje wszystkie rysunki z ekranu bez zmiany pozycji żółwia | zmaż |
| **wróć** | | „wróć na początek" | ustawia żółwia w pozycji początkowej tzn. na środku ekranu | wróć |
| **UstalGrubośćPisaka** *grubość* | **ugp** *grubość* | | ustala jaką szerokość ma mieć „rysik" pisaka np. 3 piksele | UstalGrubośćPisaka 3<br>ugp 3 |
| **UstalKolorPisaka** *nr_koloru* | **ukp** *nr_koloru* | | ustala kolor pisaka, pod numerami od 1 do 15 są poszczególne kolory<br>np. 1-niebieski, 2-zielony, itd. | UstalKolorPisaka 1<br>ukp 2 |
| **UstalTło** *nr_koloru* | | „ustal kolor tła" | ustala kolor tła, np. niebieski | ustaltło 1 |
| **Zamaluj** | | | Należy podnieść pióro, ustawić się wewnątrz figury zamkniętej i opuścić pióro. Zamaluje (wypełni) wnętrze tej figury kolorem, który jest aktualnie ustalony (poleceniem ukp) | zamaluj |
| **powtórz** *n* **[***lista_poleceń***]** | | „wykonaj to co w nawiasach n razy" | powoduje n-krotne powtórzenie listy poleceń, np. 2 razy […] | powtórz 2 [np 50 pw 90] |

**Ćw. 1** Rysowanie figur geometrycznych: (po każdym poleceniu naciskamy Enter lub wszystkie polecenia piszemy w jednym wierszu i na końcu naciskamy Enter)

a) **Kwadrat** o boku długości 100 (np 100 pw 90 np 100 pw 90 np 100 pw 90 np 100 pw 90)
b) **Prostokąt** o bokach 200 na 100
c) **Trójkąt równoboczny** o boku długości 150
d) **Sześciokąt foremny** o boku 40
e) **Pięciokąt foremny** o boku 70



**Ćw. 2** Wykonaj jeszcze raz powyższe ćwiczenia korzystając z polecenia **powtórz.**
Wszystkie rozwiązania notuj w zeszycie !!!

**Ćw. 3 – dodatkowe:** Narysuj domek, jak obok.

Zamknij program (jak normalną aplikację środowiska Windows) i nie zachowuj efektów swojej pracy.

## 3. Temat: PROCEDURY BEZ PARAMETRÓW

- Jak nazywają się polecenia wydawane żółwiowi? - procedury
- Jakie pamiętacie procedury? Np. czyścimy ekran poleceniem? - cs, lw, np
- Co musimy podać jak chcemy przesunąć żółwia do przodu? - o ile go przesuwamy
- Co musimy podać jak chcemy obrócić żółwia w lewo, prawo? - o jaki kąt go obracamy

Polecenia wydawane żółwiowi to tzw. **procedury.** Jedne z nich nie wymagają podawania parametrów, czyli dodatkowych informacji np. procedura **cs, sż, pż, pod,** itd.. Jednak inne np. **lw, pw, np, ws, ugp, ukp** wymagają podania danych, są nimi odpowiednio: kąt, odległość na jaką przesuwamy, grubość, kolor, itp.

Można samemu zdefiniować jakąś procedurę, czyli ją napisać.
Może to być procedura (polecenie), która po wywołaniu narysuje np. kwadrat o boku 60

- Jak za pomocą polecenia Powtórz narysować kwadrat o boku dł. 60?

**Powtórz 4[np 60 pw 90]**

Chcemy żeby w momencie, gdy napiszemy **kwadrat** (damy polecenie kwadrat) pojawił się nam na ekranie taki właśnie kwadrat. Musimy napisać procedurę o nazwie **kwadrat.**

Wpisujemy w wierszu ekranu tekstowego, po każdym wierszu naciskając ENTER
Budowa każdej procedury:

**oto** ***nazwa_procedury***
***Treść_procedury***
**już**

Procedura „rysująca" kwadrat:

**oto** *kwadrat*
*Powtórz 4[np 60 pw 90]*
**już**

Jak naciśniemy klawisz **F4** (lub 4 ikonkę u gór; lub menu Okno – Pokaż **pamięć**) zobaczymy naszą procedurę kwadrat wraz z jej zawartością.
Aby napisać kolejną nową procedurę możemy postępować jak wcześniej lub po otwarciu pamięci (**F4) - Obiekty – Dodaj procedurę - ... – koniec**
W pamięci edytora poruszmy się jak po zwykłym edytorze tekstowym, możemy wprowadzać zmiany (2 razy klikamy na nazwie procedury)

Aby zobaczyć czy procedura działa piszemy jej nazwę i Enter (kwadrat – Enter).

**Ćwiczenia do samodzielnego wykonania:**
Zdefiniuj procedury rysowania: (boki długości 60)

- Trójkąta równobocznego
- Pięciokąta foremnego
- Sześciokąta foremnego
- Ośmiokąt foremny

Jak narysować okrąg? Napisz odpowiednią procedurę

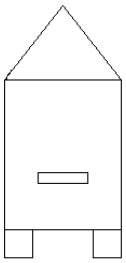
Zdefiniowane przez nas procedury znajduję się jedynie w oknie pamięci programu Logo. Aby z nich móc korzystać po ponownym włączeniu programu, musimy je zapisać w pamięci komputera (tzn. w pliku na dysku).
**Zapis pliku jako:**

- **projekt** (zawiera tekst procedur i opis wyglądu okna)
  Plik - Zapisz projekt - ...
- **program źródłowy** (tzn. same teksty procedur)
  Plik - Zapisz jako - ...

Aby po ponownym uruchomieniu programu korzystać z tych procedur, musimy otworzyć plik, w którym je zapisaliśmy (menu **Plik – Otwórz projekt - ...**)

**Zadanie domowe:**
Zastanów się z jakich elementów (figur) składa się ul. Zaproponuj wymiary jego poszczególnych części. Jakie procedury potrzebujesz do narysowania takiego rysunku? Wypisz je. Np. procedura rysująca dach:

Oto dach
pw 30
powtórz 3 [ np 100 pw 120]
lw 30
już



## 4. Temat: PROCEDURY Z PARAMETRAMI

**oto** *kwadrat*
*Powtórz 4[np 60 pw 90]*
**już**

- wprowadzenie pojęcia parametru (do powyższej procedury wprowadzimy *parametr* – którym będzie liczba określająca długość boku kwadratu)

**oto *kwadrat* :bok** ← parametr formalny
*powtórz 4[np* ***:bok*** *pw 90]*
**już**

wywołanie procedury: kwadrat **150** ← parametr aktualny

W chwili wywołania procedury w miejsce tzw. *parametru formalnego* (:bok) wstawiana jest jego wartość (np. 150) – tzw. *parametr aktualny*. W ten sposób możemy wywołać tę samą procedurę z różnymi parametrami aktualnymi, czyli rysować kwadraty o różnych długościach boków, bez konieczności pisania nowej procedury.

1. Parametry formalne – to parametry procedury występujące w jej definicji (np. :bok, :kąt, :a, :h, itp.)
2. Parametry aktualne – to wartości podawane w chwili wywołania procedury, tzn. wartości dla których dana procedura ma być wykonana (np. 100, 90, 350, itd.)

Ćwiczenia:

1. Zmodyfikuj napisane na poprzedniej lekcji procedury rysujące:
   - Trójkąt równoboczny
   - Pięciokąt foremny
   - Sześciokąt foremny
   - Ośmiokąt foremny

   tak, by można było przy ich wywołaniu określić długość boku.

2. Zdefiniuj procedurę *figura,* służącą do rysowania wielokąta foremnego o dowolnej liczbie boków i dowolnej ich długości.

Efekty pracy notuj w zeszycie!!!

(Wskazówki*:* dwa parametry formalne: **:n –** ilość kątów wielokąta, **:bok** – długość boku tego wielokąta, **360/:n** – kąt o jaki będzie się obracał żółw rysując „n-kąt" foremny)

```
oto figura :n :bok
powtórz :n [ np :bok pw 360/:n]
już
```

wywołanie: *figura 5 60* /enter pięciokąt foremny o boku długości 60

## 5. Temat: ZMIENNE I FUNKCJE W LOGO

```
oto figura :n :bok
powtórz :n [ np :bok pw 360/:n ]
już
```

1. Tworzenie zmiennej i nadawanie jej wartości:
   Polecenie:

   **przypisz "**nazwa_zmiennej wartość
   lub
   **przyp "**nazwa_zmiennej wartość

**Ćw. 1.**

```
przypisz "liczba 123
przyp "x :liczba+10
```

Nazwa miennej — wartość

Uwaga!!!
"liczba – oznacza nazwę zmiennej
:liczba – oznacza wartość zmiennej o nazwie *liczba*

2. Odczytywanie wartości utworzonych zmiennych (x i liczba)

   **pisz** :nazwa_zmiennej

   czyli:

   pisz :liczba /Enter 123
   pisz :x /Enter 133

**Ćw. 2.**

Zmodyfikuj procedurę *figura*, by kąt obrotu żółwia nie był obliczany przy każdym powtórzeniu. Zmienną przechowującą wartość kąta obrotu żółwia (360/ :n) nazwij *obrót*.

```
oto figura :n :bok
przypisz "obrót 360/:n
powtórz :n [ np :bok pw :obrót ]
już
```

**Ćw. 3.**

Zdefiniuj procedurę *figury*, służącą do rysowania wielokątów foremnych o dowolnej liczbie **boków** i **obwodzie.**

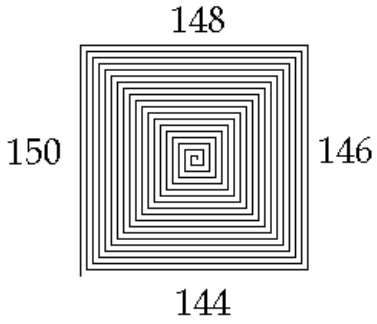
Wskazówka: (rysunek pomocniczy)

- dwa parametry (:n, :obwód)
- dwie zmienne ("obrót 360/:n, "bok :obwód/:n)

```
oto figury :n :obwód
przypisz "obrót 360/:n
przypisz "bok :obwód/:n
powtórz :n [ np :bok pw :obrót ]
już
```

**Ćw. 4.**

Napisz procedurę rysowania spirali zwijającej się do środka, jak na rysunku:




```
oto spirala
przypisz "bok 150
powtórz 75 [ np :bok pw 90 przypisz "bok :bok-2]
już
```

3. **Funkcja –** to procedura zwracająca wartość

Np. funkcja *sześcian*, której wartością jest sześcian podanej liczby:

```
oto sześcian :x
wynik :x*:x*:x
już
```

gdzie: **wynik** parametr (lub **wy** parametr) – definiowanej funkcji przypisuje wartość swojego parametru (nadaje funkcji wartość)

wywołanie np.: **pisz** *sześcian* 5 /Enter 125

## 6. Temat: PROCEDURY REKURENCYJNE

1. **Instrukcja warunkowa:**

**Jeśli** *warunek [polecenie1][polecenie2]* - jak warunek jest spełniony, to wykonane jest polecenie1, w przeciwnym wypadku - polecenie2

**Jeśli** *warunek [polecenie]* - jak warunek jest spełniony, to wykonane jest polecenie, w przeciwnym wypadku polecenie jest pomijane

**Ćw. 1.** Napisz procedurę *„większa", w wyniku której zostanie wyznaczona większa z dwóch danych liczb* ***(funkcja)***

```
Oto większa :a :b
jeśli :a>:b [wynik :a][wynik :b]
już
```

wywołanie: **pisz** *większa 12 3* */Enter 12*

2. **Procedura rekurencyjna** – to procedura wywołująca samą siebie. Żeby nie wykonywała się w nieskończoność musi zawierać **warunek zakończenia działania.**

Np.:

```
oto spiralarek :bok
jeśli :bok<=0 [stop]
np :bok
pw 90
spiralarek :bok-2
już
```

warunek zakończenia działania programu

odwołanie do samej siebie, ze zmniejszonym parametrem

**stop** – powoduje zakończenie działania procedury

3. **Ćwiczenia:**
   I. Losowe „błądzenie" żółwia po monitorze:

```
oto błądzenie :ile
jeśli :ile<1 [stop]
np losowa 10
pw losowa 360
błądzenie :ile-1
już
```

   II. Spirala o dowolnie zmieniającym się boku i kącie

```
oto spirala ::bok :kąt :dodatek
jeśli :bok>100 [stop]
np :bok pw :kąt
spirala  :bok+:dodatek  :kąt :dodatek
już
```

   wywołanie np.:

```
spirala  1     144    5
spirala  2     225    2
spirala  5     160    4
spirala  10    271    1      itp.
```

   III. Procedura obliczająca $a^n$ – **funkcja** potęga

```
oto potęga  :a  :n
jeśli :n=0 [wynik 1]
wynik :a*potęga  :a :n-1
już
```

   $a^0=1$
   $a^1=a$
   …
   $a^n=a^{n-1}a$

## 7. Temat: PROGRAMOWANIE W JĘZYKU LOGO

1. Rozwiązywany problem dzielimy na etapy. Definiujemy krótkie procedury, które możemy wywołać oddzielnie.
2. Schemat postępowania:
   - analiza problemu (rysunku)
   - sformułowanie algorytmu (metoda wstępująca i metoda zstępująca) (**ćw. w załączniku)**

3. Ćwiczenia do samodzielnego wykonania: (**kwiat_i_ul_ćw1; flagi_ćw2**)

## Ćwiczenia – załączniki do lekcji

## **WSTĘPUJĄCA METODA PROGRAMOWANIA** - „OD SZCZEGÓŁU DO OGÓŁU"

łuk

↓

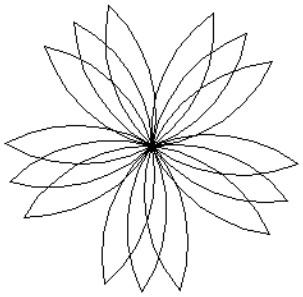
płatek

↓

trzy_płatki

↓

kwiatek

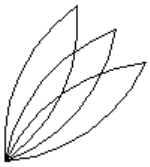


**Kwiatek**

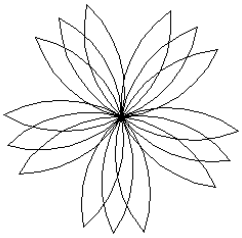
oto **łuk**
 powtórz 6 [np 20 pw 10]
już

oto **płatek**
 powtórz 2 [łuk pw 120]
już

oto **trzy_płatki**
 powtórz 3 [płatek pw 15]
już

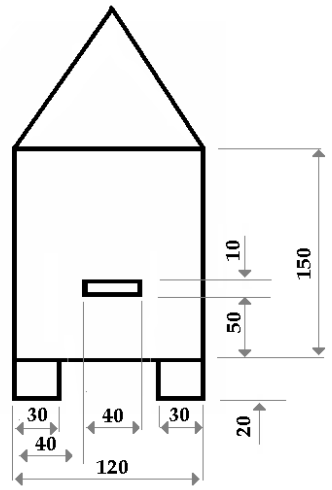


oto **kwiat**
 powtórz 5 [trzy_płatki pw 27]
już



## **ZSTĘPUJĄCA METODA PROGRAMOWANIA** - „OD OGÓŁU DO SZCZEGÓŁU"

**ul**
↓
ściana
↓
dach
↓
nogi
↓
okienko




I procedury pomocnicze (z parametrami):
- prostokąt
- trójkąt
- przesuń (przenoszenie żółwia we wskazane miejsce na ekranie)

```
oto ul
 ściana
 dach
 nogi
 okienko
już
```

```
oto ściana
 prostokąt 150 120
już
```

```
oto prostokąt :a :b
 powtórz 2 [np :a pw 90 np :b pw 90]
już
```

```
oto dach
 przesuń 0 150
 pw 30
 trójkąt 120
 lw 30
 przesuń 0 -150
już
```

```
oto trójkąt :bok
 powtórz 3 [np :bok pw 120]
już
```

```
oto przesuń :x :y
 pod
 pw 90 np :x
 lw 90 np :y
 opu
już
```

```
oto nogi
 prostokąt -20 30
 przesuń 90 0
 prostokąt -20 30
 przesuń -90 0
już
```

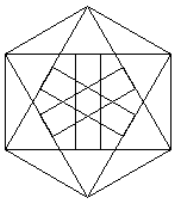
```
oto okienko
 przesuń 40 50
 prostokąt 10 40
 przesuń -40 -50
już
```

zmodyfikowana procedura *ul*

```
oto ul1
 cs
 przesuń -60 -100
 ściana
 dach
 nogi
 okienko
 przesuń 60 100
już
```

## Kwadraty

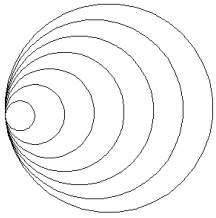
```
oto kwadrat :bok
 powtórz 4 [np :bok pw 90]
już

oto kwadraty6
 powtórz 6 [kwadrat 80 np 80 pw 60]
już
```



kwadraty6

## Pawie oczko:

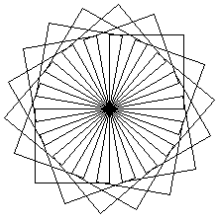
```
oto okrąg :promień
 powtórz 36 [np 0,175 * :promień pw 10]
już

oto pawie_oczko
 cs
 przypisz "promień 20
 powtórz 7 [okrąg :promień przypisz "promień :promień + 20]
już
```



pawie_oczko

## Ornament z kwadratów:

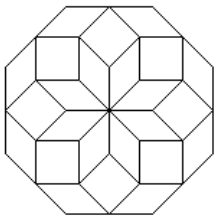
```
oto kwadrat :bok
 powtórz 4 [np :bok pw 90]
już

oto kwadraty
 cs
 powtórz 18 [kwadrat 75 pw 20]
już
```



kwadraty

## Ornament z ośmiokątów

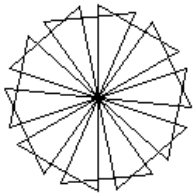
```
oto ośmiokąt :bok
 powtórz 8 [np :bok pw 45]
już

oto ośmiokąty
 cs
 powtórz 8 [ośmiokąt 30 pw 45]
już
```



ośmiokąty

## Ornament z trójkątów

```
oto trójkąt :bok
 powtórz 3 [np :bok pw 120]
już

oto trójkąty
 cs
 powtórz 10 [trójkąt 60 pw 36]
już
```

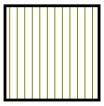


trójkąty

## Zamalowany kwadrat

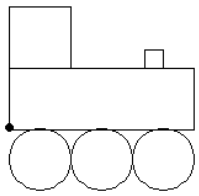
```
oto kwadrat :bok
 powtórz 4 [np :bok pw 90]
już

oto przesuń :y :x
pod
 np :y
 pw 90
 np :x
 lw 90
 opu
już
```

```
oto wzorek :grubość :kolor :wzór
 cs
 ugp :grubość
 ukm :kolor
 uwm :wzór
 kwadrat 100
 przesuń 50 50
 zamaluj
 przesuń -50 -50
już
```



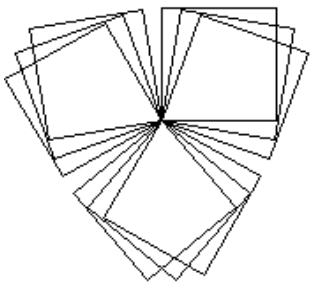
# Procedura rysowania lokomotywy

```
oto koło :promień
 powtórz 36 [np 0,175 * :promień pw 10]
już

oto kwadrat :bok
 powtórz 4 [np :bok pw 90]
już

oto lokomotywa
 prostokąt 50
 przesuń 0 50
 kwadrat 50
 przesuń 110 0
 kwadrat 15
 przesuń -110 0
 przesuń 0 -76
 koło 25
 przesuń 50 0
 koło 25
 przesuń 50 0
 koło 25
 przesuń -100 26
już

oto prostokąt :bok
 powtórz 2 [np :bok pw 90 np 3 * :bok pw 90]
już

oto przesuń :x :y
 pod
 pw 90 np :x lw 90 np :y
 opu
                                    już
```



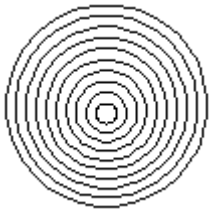
## Zadanie z kwadratami:

```
oto kwadrat :bok
 powtórz 4 [np :bok pw 90]
już

oto trzy_flagi :bok
 powtórz 3 [kwadrat :bok pw 10]
już

oto wzorek :bok
 powtórz 3 [trzy_flagi :bok pw 90]
już
```



## Zadanie z okręgami

```
oto okrąg :r
 pod lw 90 np :r pw 90 opu
 powtórz 36 [np 0,175 * :r pw 10]
 pod pw 90 np :r lw 90 opu
już

oto rysunek :r :n
 jeśli :r <= 0 [stop]
 okrąg :r
 rysunek :r - :n :n
już
```

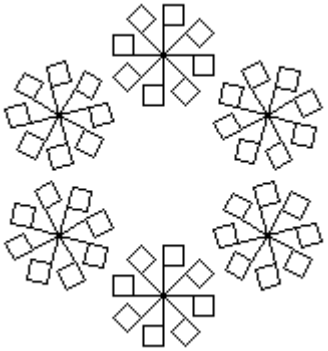


**Flagi:**

oto **kwadrat**
 powtórz 4 [np 10 pw 90]
już

oto **flaga**
 np 15 kwadrat ws 15
już

oto **wzorek**
 powtórz 8 [flaga pw 45]
już



oto **obrazek**
 powtórz 6 [pod np 60 opu wzorek pod ws 60 opu pw 60]
już

W analogiczny sposób, przygotuj odpowiednie procedury do rysowania poniższych „flag”