

# Algorytmy

**Algorytm** - uporządkowany sposób postępowania przy rozwiązywaniu problemów z uwzględnieniem opisu danych oraz opisu kolejnych czynników prowadzących do jego rozwiązania w skończonym czasie.

To szczegółowy przepis opisujący czynności, działania które powinny być wykonane przez urządzenie, aby dojść do zamierzonego celu. Każdy program działa według określonego algorytmu. Jest to przepis opisujący krok po kroku rozwiązanie problemu lub osiągnięcie jakiegoś celu.

Przykłady algorytmów:

- Przepis kuchenny
- Algorytm działania linii przemysłowej
- Algorytm znajdowania pierwiastków równania kwadratowego
- Algorytm wyznaczania najkrótszej trasy w nawigacji samochodowej

**Metody opisu algorytmów :**

- Lista kroków (metoda tekstowa)
- Graf stanów (metoda graficzna)
- Za pomocą języka programowania lub pseudokodu (uniwersalnego języka programowania)
- Schemat blokowy (metoda graficzna)
- Grafcet / SFC (metoda graficzna)

**Lista kroków** – to metoda tekstowa. Opis działania przedstawia się w postaci kolejnych kroków . To najprostsza metoda tworzenia algorytmu.

PRZYKŁAD 1

Krok 1. Zagotuj wodę.

Krok 2. Włóż jajko do gotującej się wody.

Krok 3. Odczekaj 3 minuty.

Krok 4. Wyjmij jajko z wody.

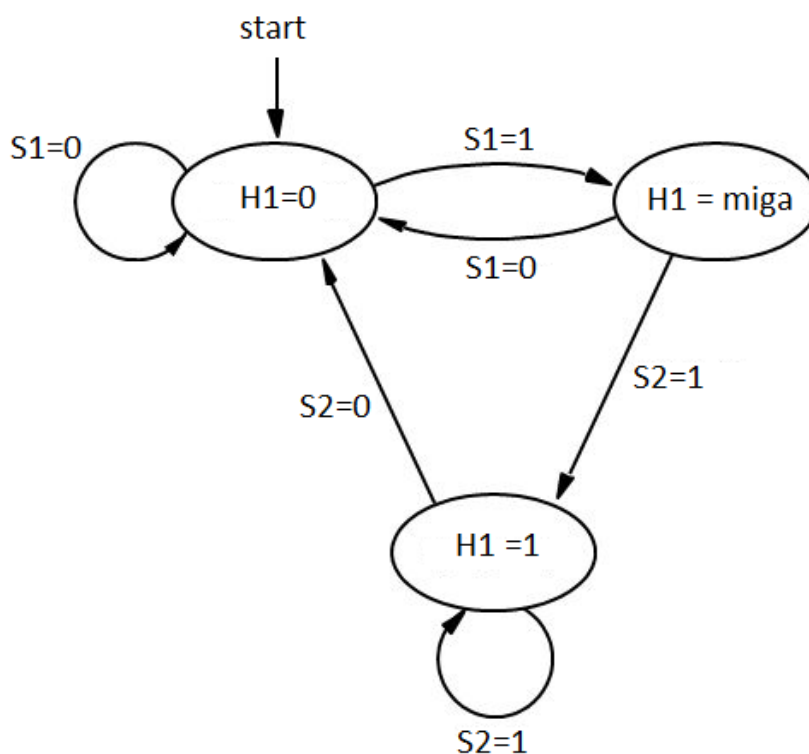
Krok 5. Odstaw wodę.

PRZYKŁAD 2

Lista kroków:

1. Początek algorytmu
2. Podaj bok a
3. Podaj bok b
4. Czy bok  $a > 0$ ? jeśli tak idź do kroku 5, jeśli nie podaj komunikat wyjściowy: "nie można obliczyć obwodu" i zakończ algorytm.
5. Czy bok  $b > 0$ ? jeśli tak idź do kroku 6, jeśli nie podaj komunikat wyjściowy: "nie można obliczyć obwodu" i zakończ algorytm.
6. Oblicz obwód  $Ob := 2 \cdot a + 2 \cdot b$
7. Wyprowadź wartość Ob
6. Koniec algorytmu

**Graf stanów** - to metoda graficzna, w której umieszcza się wszystkie możliwe stany wyjść układu i łączy się je liniami warunkowymi. Metoda ta jest popularna w przedstawianiu działania tzw. automatów.



**języka programowania lub pseudokod** - Pseudokod jest to połączenie języka naturalnego z elementami języka programowania.

Przykład 1:

Algorytm wczytuje i dodaje 10 liczb.

Pseudokod:

Start

$i := 0$

Dopóki  $i < 10$

Wczytaj(a)

$\text{Suma} := \text{Suma} + a$

$i := i + 1$

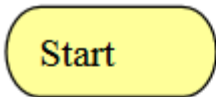
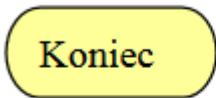
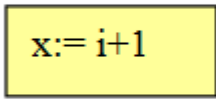
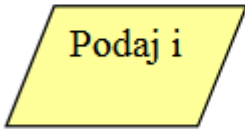
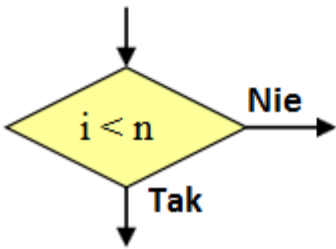
Koniec

**Schemat blokowy** - to metoda graficzna popularna głównie w informatyce. Jest to diagram, na którym algorytm jest reprezentowany przez opisane figury geometryczne, połączone liniami zgodnie z kolejnością wykonywania czynności wynikających z przyjętego algorytmu rozwiązania zadania. Pozwala dostrzec istotne etapy algorytmu i logiczne zależności między nimi. Na schemacie blokowym poszczególne operacje są opisane za pomocą bloków połączonych ze sobą strzałkami.

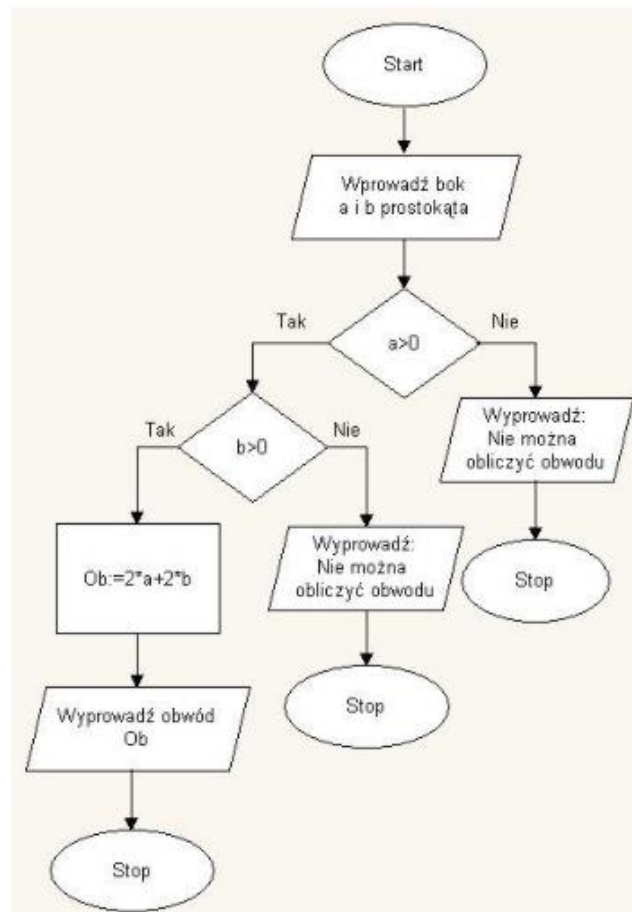
W zastosowaniach inżynierskich metoda schematu blokowego posiada zbyt wiele wad.

Najwięcej trudności sprawia wstawianie kaskadowe wielu warunków (algorytm zajmuje dużą przestrzeń).

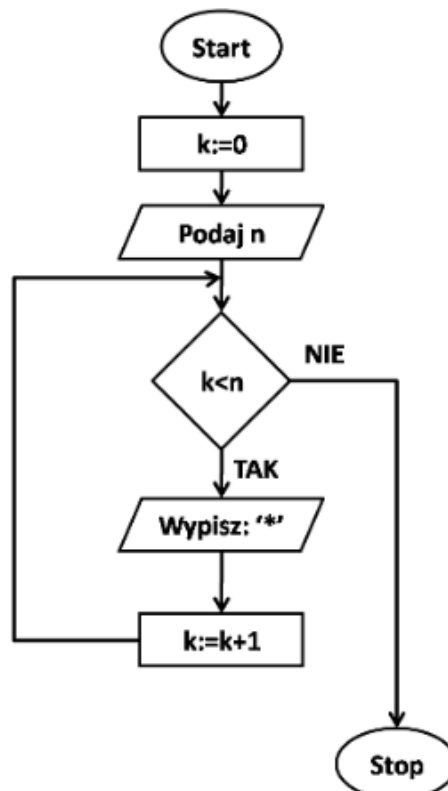
Jakakolwiek rozbudowa algorytmu wymusza jego rysowanie od nowa .

Symbol graficzny	Nazwa	Funkcja
	Blok graniczny	Oznacza początek algorytmu
	Blok graniczny	Oznacza zakończenie algorytmu
	Blok operacyjny	Służy do zapisania wykonywanych operacji na przykład działań algebraicznych, operacji podstawienia itp.
	Blok wejścia Blok wyjścia	Służy do wprowadzenia danych Służy do wyprowadzenia wyniku operacji
	Blok warunkowy	Służy do sprawdzenia warunku

### Przykład 1



### Przykład 2 - użycie licznika w celu powtórzenia sekwencji kilka razy



**GRAFCET** (fr. Graphe de Commande Etape – Transition) - jest to metoda tworzenia algorytmów pozwalających na opisanie systemu odnoszącego się do urządzenia lub procesu technologicznego pracującego sekwencyjnie tj. krok po kroku. Może być stosowany w przemysłowych systemach sterowania niezależnie od wykorzystywanej technologii: elektryczna, elektromechaniczna, pneumatyczna itp. , gdyż opisuje tylko funkcjonalne przedstawienie systemu sterowania nie opisując programowania.

Celem metody GRAFCET jest standaryzacja funkcjonalna i graficzna przedstawienia systemu sterowania. W swoich zastosowaniach diagram GRAFCET wykorzystuje ograniczoną liczbę prostych symboli i kieruje się zbiorem określonych reguł.

Diagram GRAFCET składa się z:

- kroków (steps) - które zawierają działania;
- przejść (transitions ) - które zawierają warunki;
- połączeń pomiędzy krokami i przejściami .

Sposób funkcjonowania algorytmu sterowania Grafcet i SFC zdefiniowany jest następującymi regułami:

### **Reguła 1**

Przejście jest realizowane tylko wówczas, gdy krok bezpośrednio przed nim jest aktywny i warunek przejścia jest spełniony .Realizacja przejścia powoduje dezaktywację kroku (kroków) który wcześniej był aktywny i aktywację kroku (kroków) do którego prowadzi przejście ,którego warunek przejścia został spełniony.

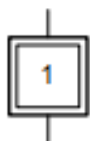
### **Reguła 2**

Kroki i przejścia muszą występować naprzemiennie tworzy sekwencję diagramu (nie mogą wystąpić po sobie 2 kroki lub 2 warunki).

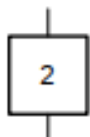
### **Reguła 3**

Warunki sprawdzane i realizowane są w kolejności od lewej do prawej (jeśli jednocześnie spełnione są 2 warunki – zostanie zrealizowany ten z lewej strony). Warunek o wyższym priorytecie umieszcza się zawsze z lewej strony.

**Symbole graficzne algorytmu GRAFCET i SFC :**



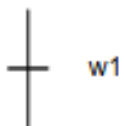
Krok początkowy



Krok



Makro ( krok )



Przejście



Działanie powiazane z krokiem



Początek koniunkcji



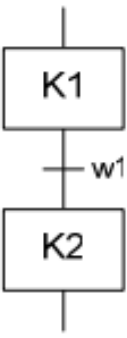
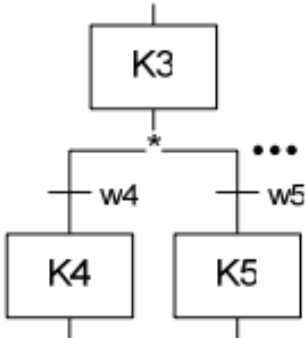
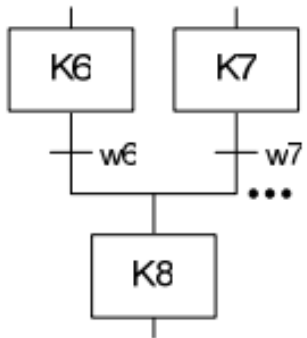
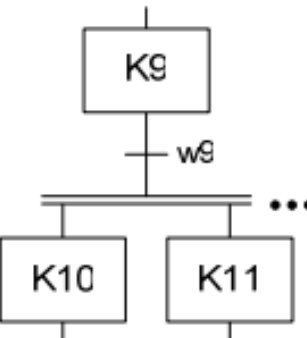
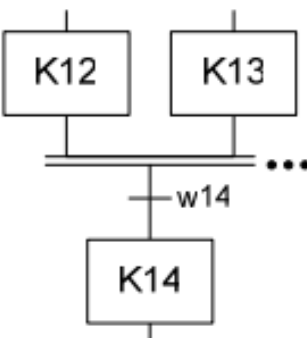
Koniec koniunkcji

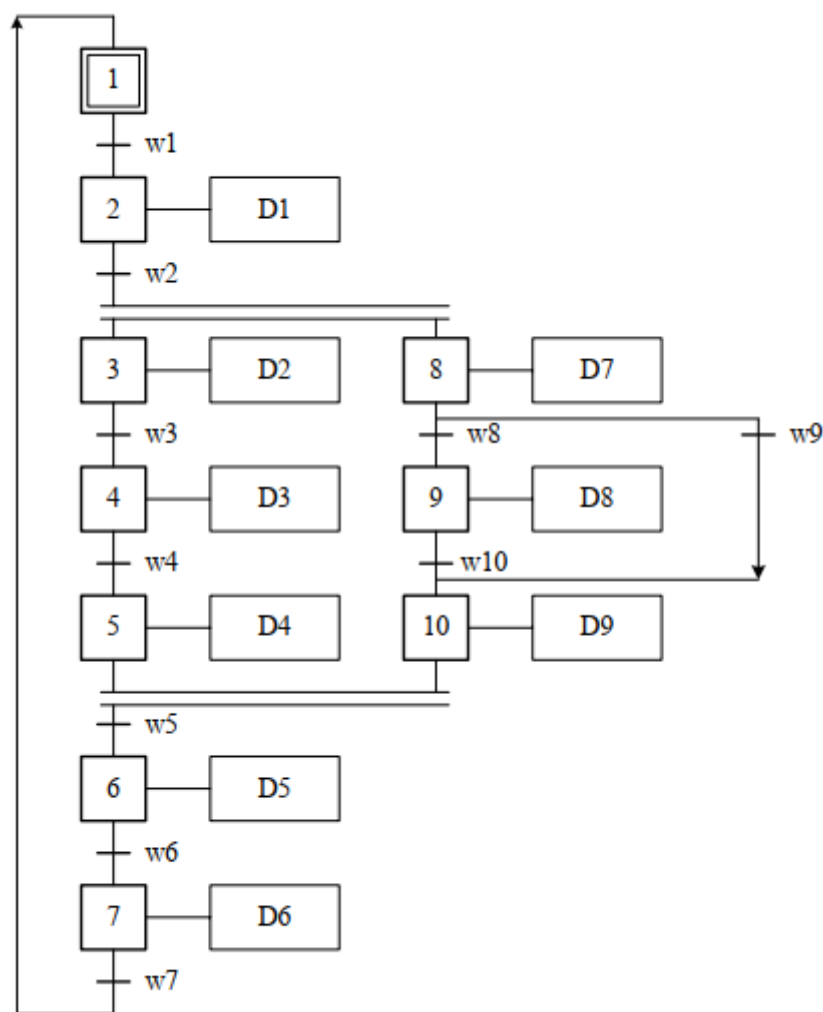


Początek alternatywy



Koniec alternatywy

Przykład	Opis sekwencji
	<p><b>Sekwencja pojedyncza – prosta</b></p> <p>Kroki i przejścia występują kolejno po sobie. Przejście z kroku K1 do K2 nastąpi, jeżeli krok K1 będzie aktywny i jednocześnie zostanie spełniony warunek przejścia w1 (w1=TRUE)</p>
	<p><b>Sekwencja rozbieżna</b></p> <p>Wybór pomiędzy różnymi sekwencjami odbywa się na zasadzie sprawdzenia warunków przejścia, w kolejności od lewej do prawej. Spotyka się również struktury z numeracją kolejności odpytywania. Przejście z K3 do K5 nastąpi, jeśli K3 będzie aktywny oraz warunek przejścia w4 nie będzie spełniony, a będzie spełniony warunek w5.</p>
	<p><b>Sekwencja zbieżna</b></p> <p>Wszystkie sekwencje rozbieżne kończą się symbolami przejścia. Przejście do kroku K8 nastąpi, jeśli K6 będzie aktywny oraz warunek przejścia w6 będzie spełniony, albo K7 będzie aktywny i jednocześnie będzie spełniony warunek w7.</p>
	<p><b>Sekwencja równoczesna rozbieżna</b></p> <p>W tym przypadku następuje rozdzielenie procesu na dwie lub więcej równoczesnych sekwencji. Przejście z kroku K9 do kroków K10 i K11 nastąpi, jeśli K9 będzie aktywny oraz warunek przejścia będzie spełniony.</p>
	<p><b>Sekwencja równoczesna zbieżna</b></p> <p>Wszystkie sekwencje rozbieżne równoczesne kończą się symbolem podwójnej linii. Przejście do kroku K14 nastąpi, jeśli wszystkie kroki przed podwójną linią będą aktywne (K12 i K13) oraz będzie spełniony warunek przejścia w14.</p>



Przykładowy diagram języka GRAFCET  
(oznaczenia: 1÷10 – kroki, w1÷w10 – przejścia, D1÷D9 – działania powiązane z krokami)

W tym diagramie jest w sumie 10 kroków (w tym jeden krok początkowy oznaczony numerem 1) oraz 10 przejść zawierających warunki przejść oznaczone od w1 do w10.

Z założenia połączenia pomiędzy krokami przebiegają z góry do dołu diagramu; w przypadkach nieoczywistych lub innych połączeniach, kierunek połączenia może być oznaczony strzałką. Na rysunku dwa takie połączenia oznaczono strzałkami.

Z krokami są powiązane określone działania oznaczone od D1 do D9. Działania te są wykonywane wówczas, gdy krok z którym są powiązane jest aktywny, i przestają być wykonywane wówczas, gdy krok przestaje być aktywny.

Możliwość aktywacji kroków zależy od przejść je poprzedzających i aktualnej aktywności kroków w diagramie. Przykładowo, jeśli aktywny jest krok 1 i spełniony jest warunek przejścia w1 to nastąpi przejście do kroku 2, czyli dezaktywacja kroku 1 i aktywacja kroku 2. Po spełnieniu warunku przejścia realizacja przejścia jest obligatoryjna i natychmiastowa.

W diagramie oprócz połączeń prostych (typu: krok – przejście – krok – przejście itd.) są również połączenia wielokrotne.

Gdy aktywny jest krok 2 i spełniony jest warunek w2 znajdującego się za nim przejścia, to nastąpi dezaktywacja kroku 2 i równoczesna aktywacja kroków 3 i 8 (sekwencja równoległa) co na diagramie pokazane jest równoległymi liniami po warunku w2. Wyjście z tej sekwencji równoległej nastąpi wówczas, gdy będą aktywne kroki 5 i 10 oraz spełniony będzie wspólny warunek przejścia w5. Nastąpi wówczas dezaktywacja kroków 5 i 10 oraz aktywacja kroku 6.



Gdy aktywny jest krok 8, to kolejnym krokiem aktywnym może stać się albo krok 9 albo krok 10. Gdy spełniony będzie warunek przejścia w8 to nastąpi dezaktywacja kroku 8 i aktywacja kroku 9. Natomiast, gdy spełniony będzie warunek przejścia w9, to nastąpi dezaktywacja kroku 8 i aktywacja kroku 10 (krok 9 i działania z nim powiązane są omijane).

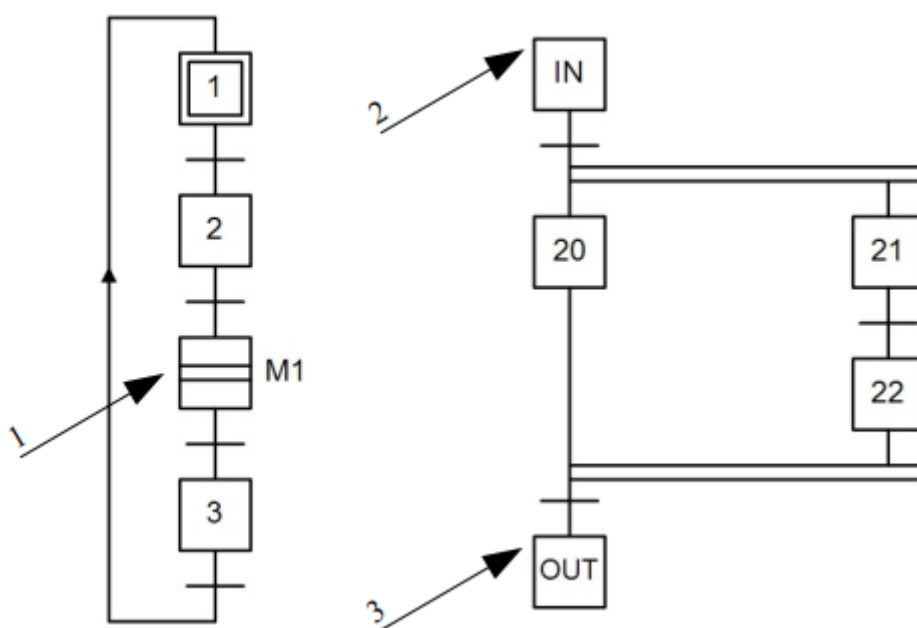
Ponieważ jest to wybór jednego kroku z dwóch możliwych to, gdy spełnione są oba warunki: w9 i w8 zostanie zrealizowany ten umieszczony po lewej stronie –aktywacja kroku 9

Jak widać z powyższego, w diagramie GRAFCET w danej chwili może być aktywny więcej niż jeden krok. W przykładzie aktywne mogą być równocześnie dwa kroki (jeden spośród kroków 3, 4, 5 i jeden spośród kroków 8, 9, 10).

Podczas trwania sekwencji równoczesnej nie można wykonać skoku z jednej tylko gałęzi (do sekwencji wchodzi się jednym wspólnym warunkiem i wychodzi również jednym warunkiem)

Gdy algorytm jest długi i skomplikowany można podzielić go na mniejsze części tworząc tzw. makro krok.

Takie połączone kroki i przejścia powinny być funkcjonalnie powiązane, tworząc w sumie złożone zadanie (funkcję) sterowania. Makro krok zawsze rozpoczyna się i kończy krokiem, i dzięki temu może być umieszczony w diagramie GRAFCET tak samo jak zwykły pojedynczy krok z poprzedzającym go przejściem i z następującym po nim przejściem. Dzięki wykorzystaniu makro kroków można osiągnąć większą przejrzystość diagramu. Diagram taki odzwierciedla ogólną strukturę sterowania, podczas gdy szczegóły ukryte są w makro krokach (możliwe jest zagnieżdżanie – tj. umieszczanie makro kroków w makro krokach).



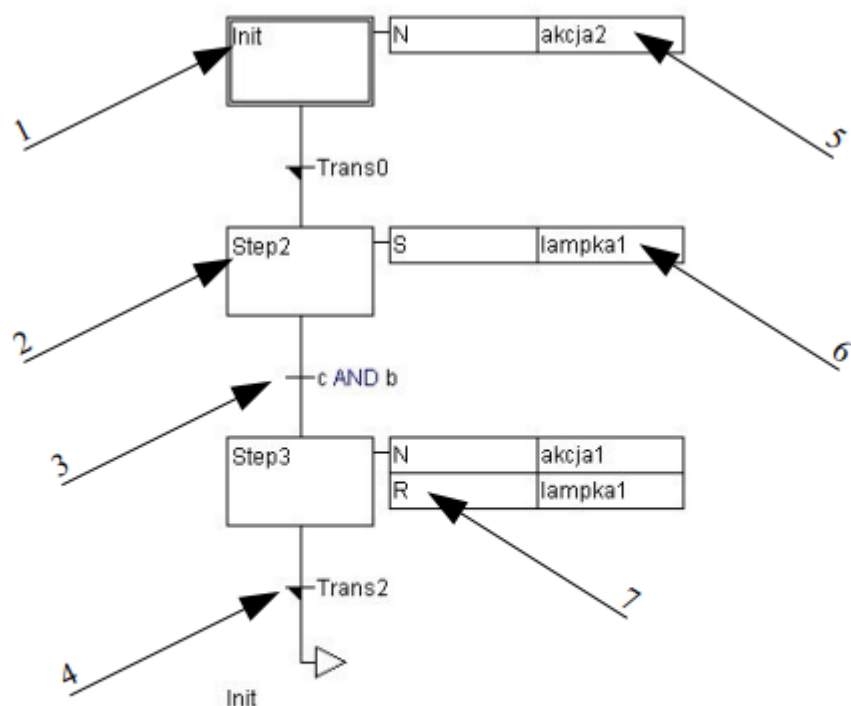
Po lewej – algorytm z makro krokiem, po prawej – zawartość makro kroku

**GRAFCET jest jedynie metodą tworzenia algorytmów i posiada dużą swobodę w kwestii opisów warunków i akcji. Działanie powiązane z krokiem może zawierać opis słowny (np. wysuń siłownik A1)**

**SFC** (ang. Sequential Functional Chart – sekwencyjny schemat funkcjonalny) – to język programowania sterowników PLC opisany w normie IEC 61131-3. Język graficzny SFC został oparty na algorytmie GRAFCET.

Tworząc język programowania wyeliminowano dowolność z opisów warunków przejścia i akcji.

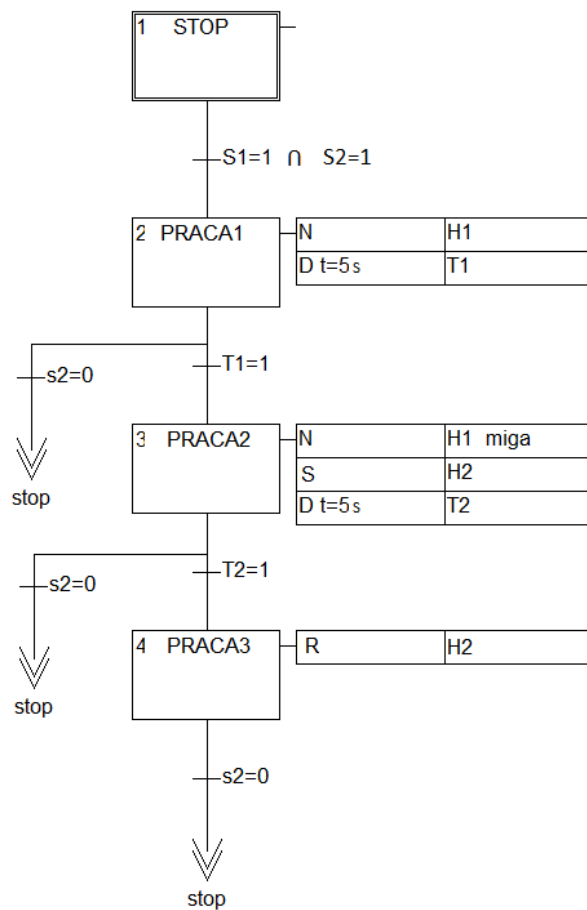
Warunki muszą stanowić funkcję logiczną z wynikiem 0 lub 1. Akcje powiązane z krokiem przyjmują wartości 0 lub 1 i opisane są za pomocą tzw. kwalifikatorów.



Przykładowy algorytm zapisany za pomocą sieci SFC; 1-krok inicjalizujący; 2-krok właściwy; 3-warunek tranzycji zapisany bezpośrednio; 4-warunek tranzycji zapisany w programie Trans2; 5-akcja zdefiniowana w programie o nazwie akcja2; 6-akcja zdefiniowana poprzez bezpośrednie odwołanie do zmiennej lampka1; 7-kwalifikatory akcji

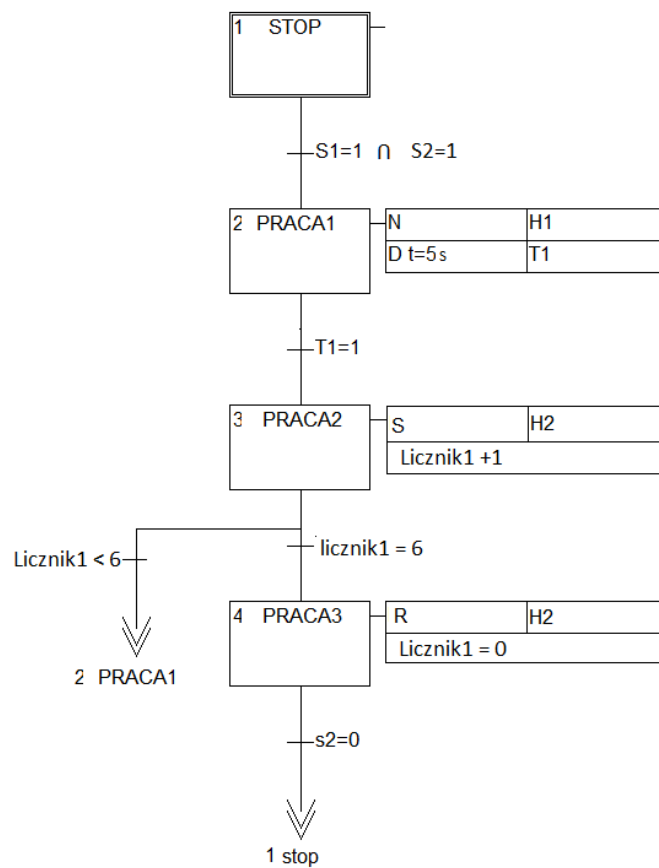
Kwalifikator	Znaczenie	Opis
N	chwilowy (ang. <i>Non-stored</i> )	akcja jest aktywna tak długo jak aktywny jest krok
R	kasowanie/reset (ang. <i>overriding Reset</i> )	akcja jest deaktywowana (reset) gdy krok jest aktywny
S	zapis/set (ang. <i>set-Stored</i> )	akcja staje się aktywna gdy krok jest aktywny i trwa do momentu deaktywacji (reset)
L	ograniczona czasowo (ang. <i>time Limited</i> )	akcja jest aktywna przez określony czas od momentu aktywacji kroku (maksymalnie przez czas aktywności tego kroku)
D	opóźniona czasowo (ang. <i>time Delayed</i> )	akcja staje się aktywna po określonym czasie od momentu aktywacji kroku jeśli krok jest nadal aktywny i pozostaje aktywna tak długo jak krok jest aktywny
P	impuls (ang. <i>Pulse</i> )	akcja wykonywana jest tylko jeden raz gdy krok staje się aktywny
SD	zapisana i opóźniona czasowo (ang. <i>Stored and time Delayed</i> )	akcja staje się aktywna po upływie określonego czasu i trwa do momentu deaktywacji (reset)
DS	opóźniona i przechowywana (ang. <i>Delayed and Stored</i> )	akcja staje się aktywna po upływie określonego czasu pod warunkiem aktywności kroku i trwa do momentu deaktywacji (reset)
SL	zapisana i ograniczona czasowo (ang. <i>Stored and time Limited</i> )	akcja jest aktywna przez określony czas od momentu aktywacji kroku

Kwalifikatory L, D, SD, DS i SL wymagają podania wartości czasu w postaci "t=5s", umieszczanego za kwalifikatorem.



Przykładowy program SFC (zastosowano pewne uproszczenie dopisując H1 miga )

Nie istnieje kwalifikator opisujący licznik. Wykonanie fragmentu programu kilka razy można zobrazować w następujący sposób:



### **Zalety i wady języka Grafset i SFC**

Ze względu na zbliżone pochodzenia obydwu metod i języków zarówno ich zalety jak i wady są podobne. Do zalet zaliczyć można prostotę i intuicyjność pisania programu.

Po stronie wad liczba cech jest zdecydowanie większa.

Przed wszystkim maksymalna liczba kroków i tranzycji zależy od implementacji (sterownika PLC) i jest zdecydowanie mniejsza niż w przypadku LD, FBD, IL czy ST.

Po drugie, implementacja opracowanego programu możliwa jest jedynie w sterownikach PLC dysponujących tym językiem, który nadal nie jest popularny.

Nie istnieje prosty sposób tłumaczenia SFC na pozostałe języki z normy (LD, IL, FBD, ST )