za<pro/>gramowani.com

</> webmaster

</> kontakt

E-mail: zaprogramowani@filiposinski.com

Discord: https://tiny.pl/7k6jp

GitHub: github.com/zaprogramowaniFO

</> Zajęcia **nr 8**

2023/05/11

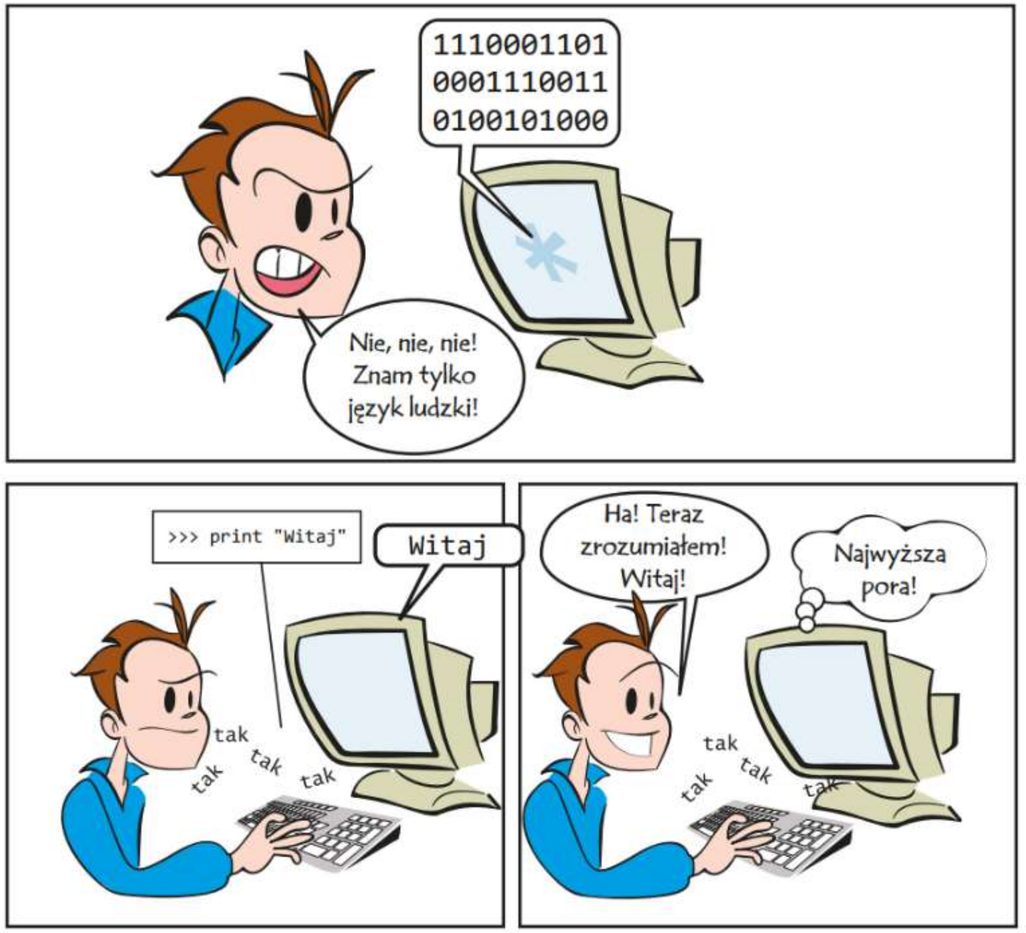
</> czym jest **programowanie**

sposób porozumiewania się człowieka z maszyną;
proces **projektowania i tworzenia** programów;
proces **testowania i naprawiania kodu** programów;

instruowanie maszyny co ma robić;
rozwiązywanie zagadek i problemów;
ulepszanie świata ;)


Swedish
Italian
German
Arabic
Japanese
Polish
Portuguese
Kurdish
Dutch
Croatian-Serbian
Catalan
Finnish
Greek
French
Chinese
Latin
Korean
Russian
Spanish
Danish
Norwegian
English
Turkish


Java
JavaScript
Python
Objective-C
Perl
Ruby
Haskell
PHP
Pascal
C++
C
Visual Basic
Transact-SQL
Clarion
Logo
Prolog
ColdFusion
Forth
PL/SQL
Ada
Lua
Yacc
TCL
Delphi
MATLAB
ActionScript
PostScript
FoxPro
ABAP
OpenEdge ABL
Scala
F#
ML
Erlang
C#
D
COBOL
Assembly
Scratch
Scheme
R
Q
Go
Fortran
Mathematica
Groovy
Tcl
Haskell
Lisp
Swift
Dart
cT
Max/MSP
SAS
PL/I


1110001101
0001110011
0100101000
Nie, nie, nie!
Znam tylko
język ludzki!
>>> print "Witaj"
Witaj
tak
tak
tak
tak
tak
Ha! Teraz
zrozumiałem!
Witaj!
Najwyższa
pora!
tak
tak
tak
tak
tak

wszystkie języki programistyczne mają wspólne założenia, które mają swój początek w latach 70’tych, takie jak:

instrukcje warunkowe, wyboru, iteracyjne;
funkcje;
klasy i obiekty;

jest to logika programowania.

</> ALGORYTMY

Czym są **algorytmy ?**

To skończony ciąg czynności, który umożliwia przekształcić dane wejściowe na dane wyjściowe;

To opis rozwiązanie danego problemu krok po kroku;

Etapy konstruowania algorytmu

1. Ustalenie problemu do rozwiązania dla algorytmu
2. Określenie danych wejściowych
3. Określenie wyniku oraz sposobu w jaki zostanie zaprezentowany
4. Wybranie metody wykonania zadania, która wg nas jest najlepsza
5. Zapisanie algorytmu
6. Sprawdzenie poprawności rozwiązania
7. Przeprowadzenie testów dla różnych danych wejściowych
8. Praktyczna ocena skuteczności algorytmu

PRZYKŁADEM ALGORYTMU MOŻE BYĆ PRZEPIS KULINARNY, NP. NA KANAPKĘ! DANYMI WEJŚCIOWYMI SĄ SKŁADNIKI, WYKONYWANYMI OPERACJAMI SĄ WSZYSTKIE CZYNNOŚCI W TRAKCIE PRZYGOTOWYWANIA KANAPKI, A DANYMI WYJŚCIOWYMI JEST GOTOWA KANAPKA! :)

Sposoby zapisania algorytmu

- opis słowny
- lista kroków
- pseudokod
- schemat blokowy - graficzna prezentacja algorytmu
- program komputerowy - zapis w danym języku programowania

</> RODZAJE ALGORYTMÓW

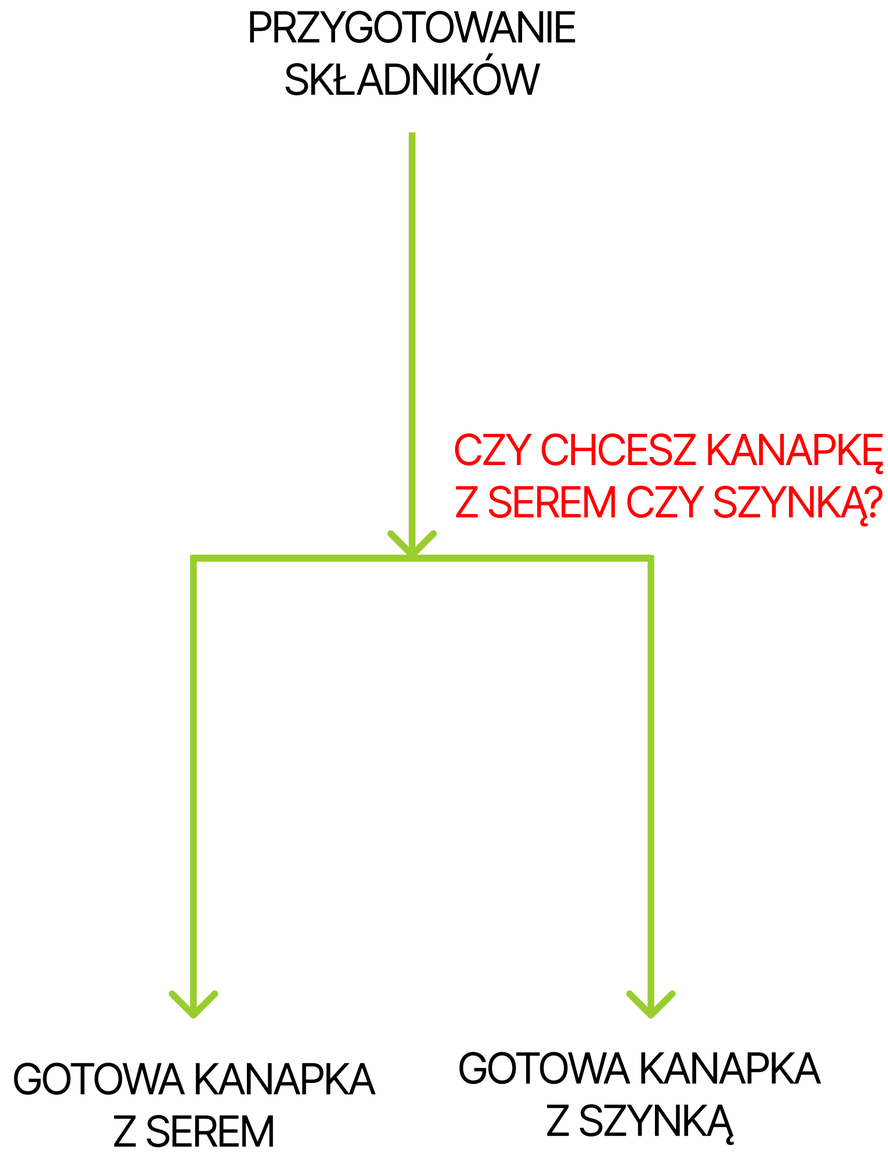
Algorytmy liniowe

To takie algorytmy, w których nie określono żadnych warunków, a każde z poleceń wykonywane jest bezpośrednio jedno po drugim. Takie algorytmy nazywamy również jako sekwencyjne.

JEST TO NAJPROSTRZY ALGORYTM. ROZPATRUJĄC PRZYKŁAD KANAPKI, NA POCZĄTKU PRZYGOTOWUJEMY SKŁADNIKI, NASTĘPNIE ROBIMY KANAPKĘ I NA KOŃCU JĄ JEMY ;)

Algorytmy warunkowe

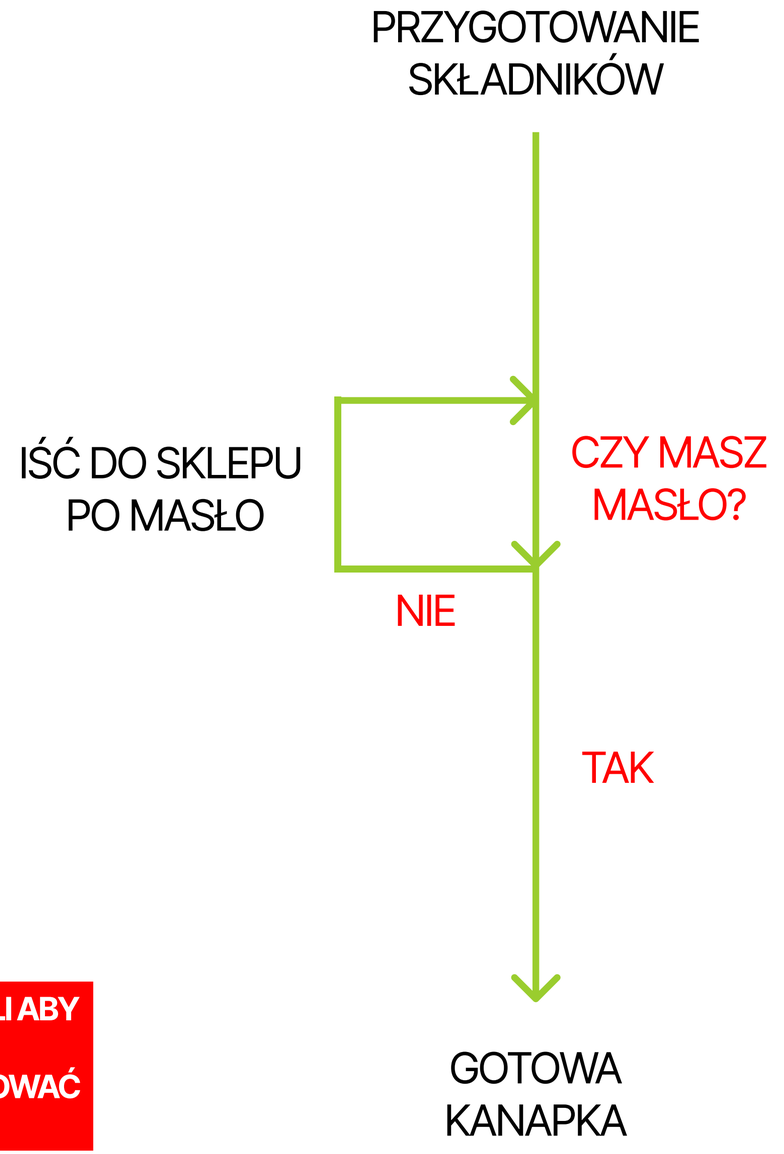
To algorytmy w których zostają postawione pewne warunki, i wykonanie konkretnych instrukcji jest uzależnione od ich spełnienia bądź nie.



ALGORYTM STAWIA JAKIŚ WARUNEK, PYTANIE KTÓRE W ZALEŻNOŚCI OD ODPOWIEDZI POPROWADZI NASTĘPNE DZIAŁANIA JEDNĄ Z DWÓCH MOŻLIWYCH ŚCIEŻEK.
W PRZYKŁADZIE KANAPKI, WARUNKIEM MOŻE BYĆ ZADECYDOWANIE CZY WOLIMY KANAPKĘ Z SZYNKĄ CZY Z SEREM.

Algorytmy iteracyjne

To algorytmy w których występuje powtarzanie pewnych operacji. Liczba powtórzeń może zostać z góry narzucona albo zależeć od spełnienia jakiegoś warunku, który za każdym powtórzeniem będzie sprawdzany.
Instrukcję powtarzania danego ciągu operacji nazywamy ITERACJĄ lub PĘTLĄ



POSTAWIONY WARUNEK, DOPÓKI NIE ZOSTANIE SPEŁNIONY, DOPÓTY NIE POZWOLI ABY WYKONANO KOLEJNE ZADANIA.
W PRZYKŁADZIE KANAPKI, DOPÓKI NIE MAMY MASŁA, TO NIE MOŻEMY PRZYGOTOWAĆ KANAPKI.

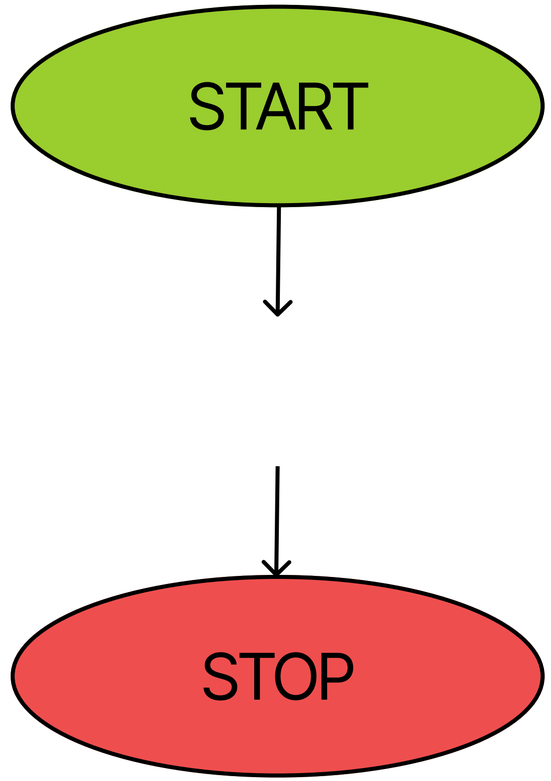
</> SCHEMATY BLOKOWE

Czym są schematy blokowe?

- Schematy blokowe umożliwiają przedstawienie algorytmu w postaci symboli graficznych, które opisują wszystkie wykonywane operacje w programie wraz z ich kolejnością.
- Pomagają zrozumieć problematykę danego zadania do rozwiązania.
- Pomagają w rozwiązaniu skomplikowanych zagadnień.

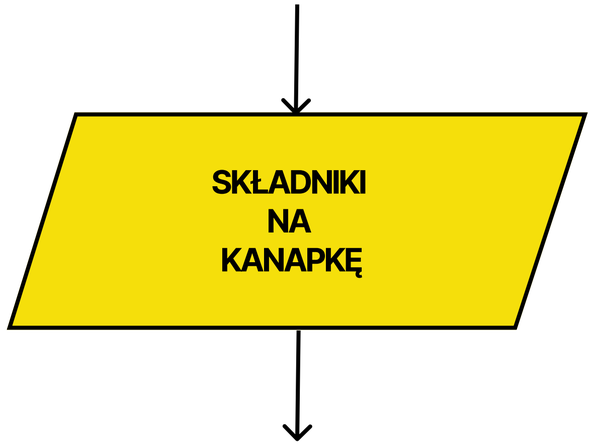
Bloki graniczne

- Bloki te oznaczają początek oraz koniec danego algorytmu. Mają kształt owalu.
- Każdy algorytm ma tylko jeden blok START, którego może wychodzić tylko jedno połączenie.
- Każdy algorytm może posiadać kilka bloków STOP (co najmniej jeden).



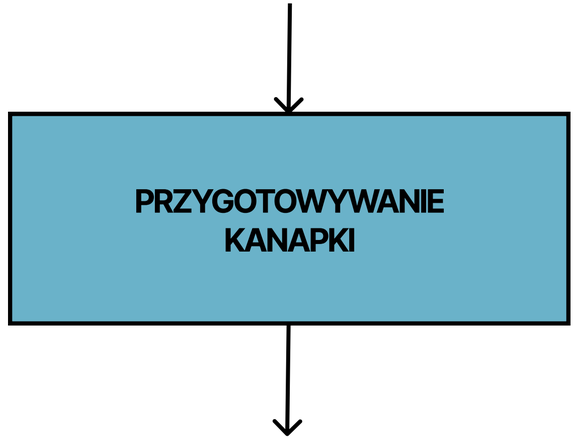
Blok wejścia / wyjścia

Blok ten odpowiada za wykonanie operacji wprowadzenia lub wyprowadzenia danych, wyników czy komunikatów.
Ma kształt podłużnego równoległoboku.
Może posiadać tylko jedno połączenie wejściowe oraz jedno połączenie wychodzące.
W jednym schemacie może znajdować się wiele takich bloków.



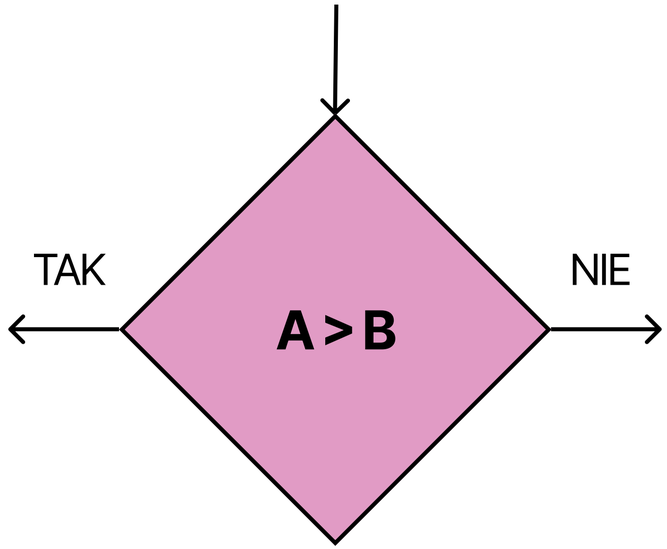
Blok operacyjny

Blok odpowiedzialny za wszelkie operacje w programie, które zmieniają wartości zmiennych, np. obliczenia.
Bloki operacyjne posiadają kształt prostokąta.
Jeden blok może zawierać więcej niż jedną operację.
Może posiadać tylko jedno połączenie wejściowe oraz jedno wychodzące.
W jednym schemacie może znajdować się wiele takich bloków.



Blok decyzyjny 1

Blok ten pozwala na wybranie jednej z dwóch możliwych ścieżek działania.
Ma kształt rombu.
Posiada tylko jedno wejście.
Posiada dwie ścieżki wychodzące:
TAK - jeśli wpisany warunek został spełniony
NIE - jeżeli wpisane warunek nie został spełniony

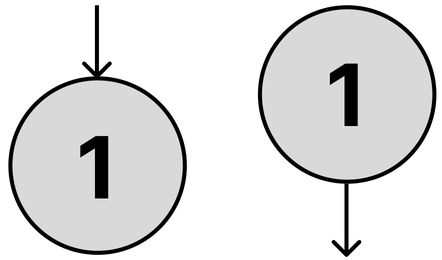


Połączenie bloków

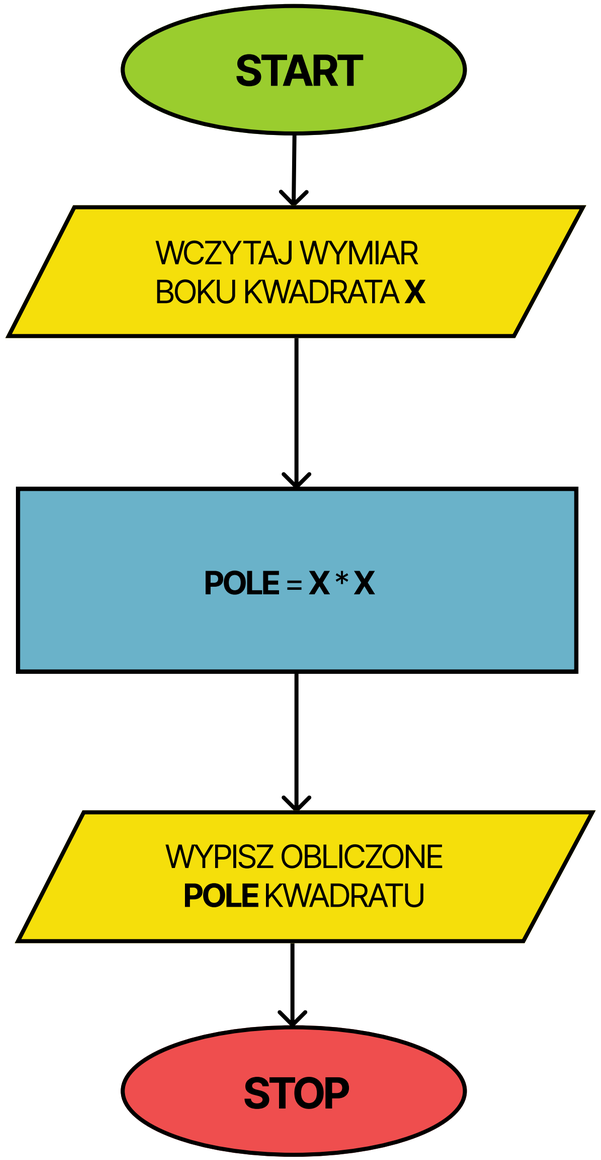
Połączenie odpowiedzialne jest za łączenie ze sobą bloków w schemacie. Określa kierunek w którą stronę przebiega przepływ danych czy też kolejność wykonywanych działań. Może też się łączyć z innymi połączeniami w jedno.

Łącznik schematów

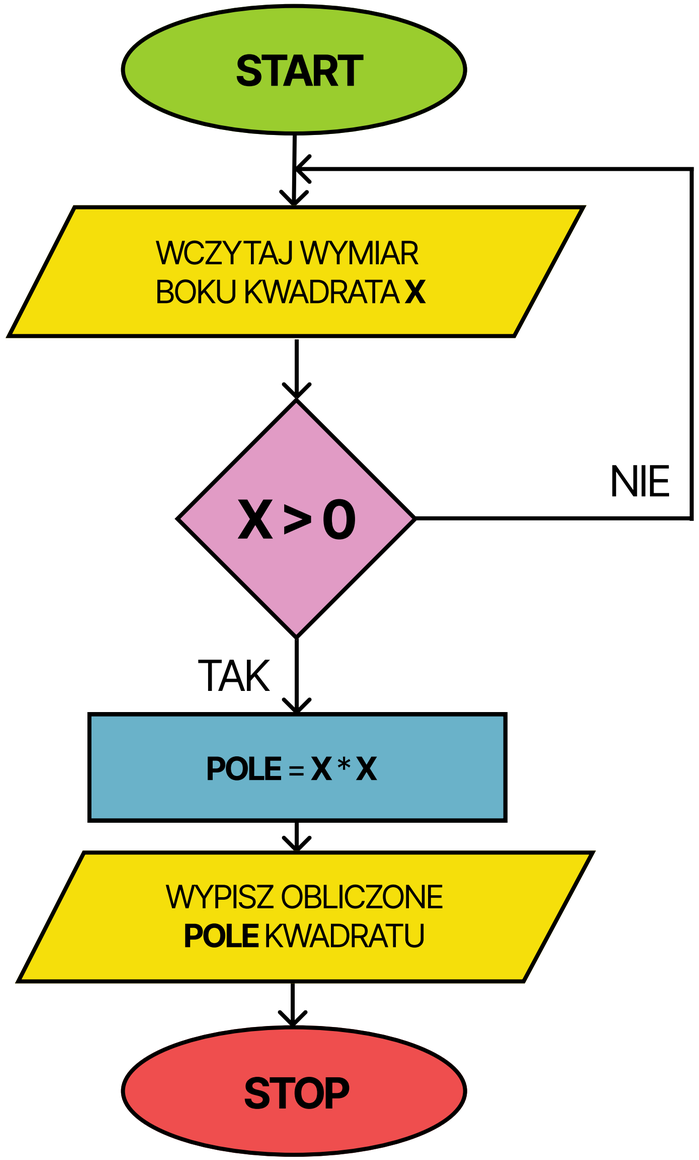
Stosujemy je, kiedy schemat blokowy chcemy narysować w kilku częściach. Jest to odsyłacz do innego fragmentu. Umieszczone oznaczenie wewnątrz musi być identyczne w obu częściach.

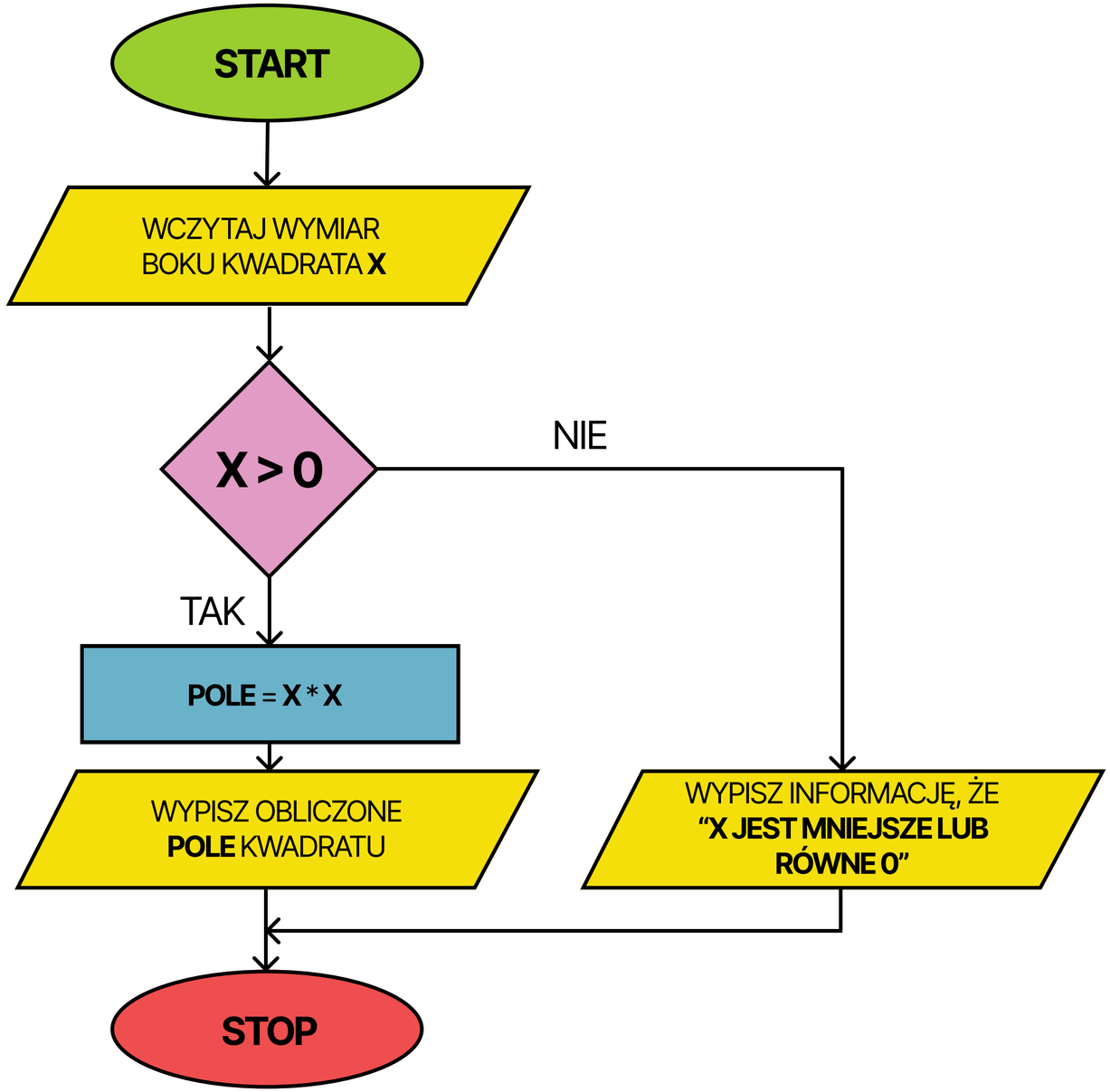


</> PRZYKŁADOWE SCHEMATY

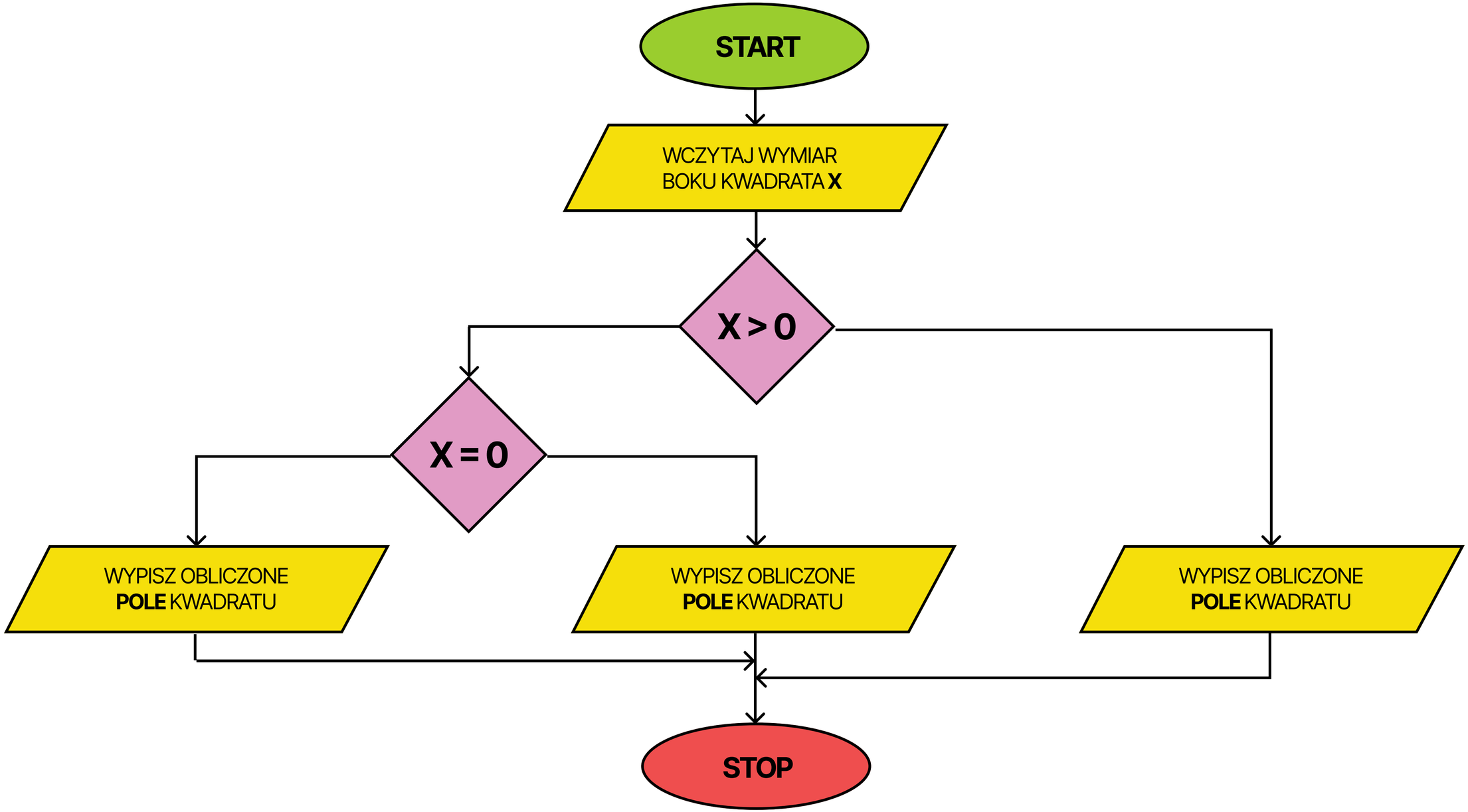


Algorytm na obliczenie pola kwadrata





Algorytm na obliczenie pola kwadratu, pod warunkiem boku większego niż 0

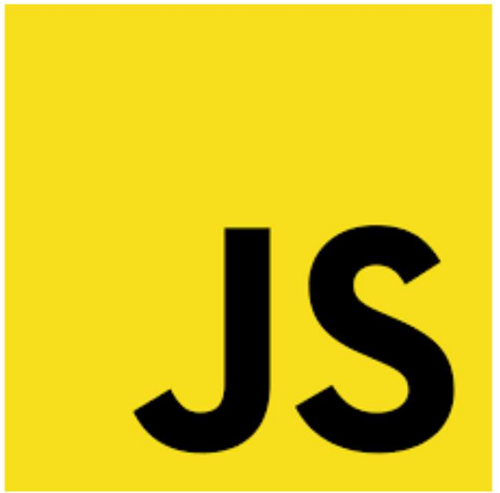


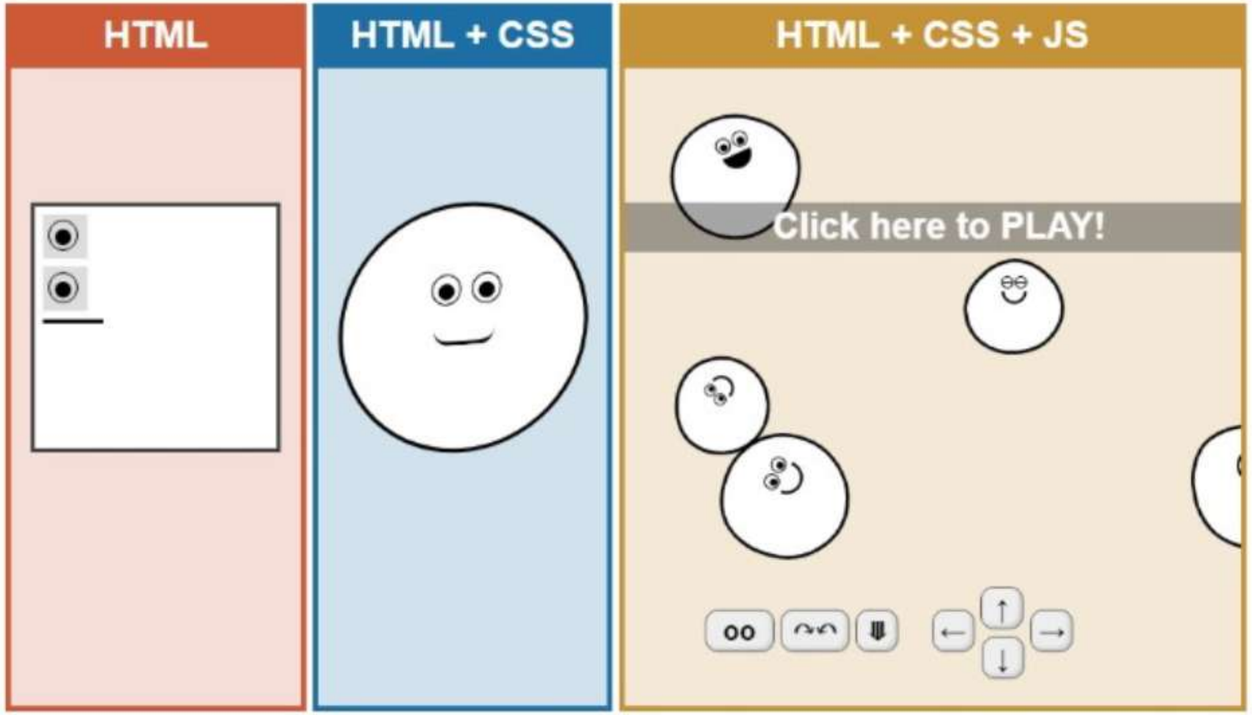
Algorytm sprawdzający czy X jest większy, mniejszy lub równy 0

</> JAVASCRIPT

JavaScript

Jest to skryptowy język programowania.
Wykorzystywany podczas pisania stron internetowych.
Pozwala na stworzenie interakcji pomiędzy użytkownikiem, a stroną internetową.




HTML
HTML + CSS
HTML + CSS + JS
Click here to PLAY!

</> Java != JavaScript

Java

- Bardziej backend
- Różne aplikacje
- Do pisania potrzebujemy JDK
- Kod musi być skompilowany
- Kod jest ukryty po skompilowaniu
- Samodzielny język
- Język silnie typowany

JavaScript

- Bardziej frontend
- Głównie aplikacje webowe
- Do pisania wystarczy notatnik
- Kod od razu gotowy
- Kod jest dostępny dla każdego
- Do działania potrzebny HTML
- Język słabo typowany

Zasady tworzenia arkusza oraz **przypięcia go do projektu**

```
<script src="ścieżka pliku"></script>
```

Aby utworzyć arkusz JS i móc go używać należy:

1. Utworzyć nowy arkusz o rozszerzeniu **.js**
2. Arkusz należy “przypiąć” do pliku strony z rozszerzeniem **html**

Zrobimy to dodając odpowiednią linijkę do naszego kodu strony na samym końcu tuż przed znacznikiem **</body>**

Podobnie jak **CSS**, skrypt możemy pisać bezpośrednio w pliku **HTML**, pomiędzy znacznikami **<script> </script>**, ale nie jest to poprawna praktyka.

</> PODSTAWOWE FUNKCJE

Podstawowe funkcje

`alert('Hello World!');` funkcja, która wywołuje okno alertu na stronie

`console.log('Hello World!');` funkcja, która drukuje w konsoli treść zawartą w argumencie

Poniższe funkcje dziś już nie stosowane:

`prompt('tekst');` funkcja, która wywołuje okno z możliwością wprowadzenia treści, np. spytać użytkownika o imię, i przypisać do zmiennej

`confirm('tekst');` funkcja, która działa podobnie do alert, ale zwraca wartość logiczną, którą można przypisać do zmiennej i później wykorzystać

</> ZMIENNE

Zmienne w JavaScript

Zmienna to **“pudełko”,** które przechowuje wartości/informacje do niej przypisane.

Każda stworzona zmienna musi mieć swoją unikatową nazwę (identyfikator).

JavaScript to język słabo typowany.

Do zmiennej możemy przypisać **dowolny typ danych**, w zależności od potrzeb.

Mogą to być np. **liczby całkowite, zmiennoprzecinkowe, teksty, wartości logiczne.**

Typy zmiennych

```
//inicjowanie zmiennej

typ nazwa = wartość;
```

const - zmienna stała, nie można później zmienić wartości;

let - zmienna do których można przypisywać wielokrotnie różne wartości;

var - zmienna ogólna, dziś już nie używana ale spotykana, podobna do let;

Nazwa ma znaczenie

Wielkość liter ma znaczenie;
Nazwa zmiennej nie może zaczynać się od cyfry;
Nazwa zmiennej nie może zawierać spacji, kropki, przecinka, myślnika;
Nazwa zmiennej nie może być "słowem kluczowym" w JavaScript

Nazywaj zmienne tak aby nazwa zmiennej miała sens i odnosiła się do wartości
(np. zmienna liczbowa **"number",** a nie **"aaaaa";**

Używaj zasady **camelCase;**

Dobrą praktyką jest **nazywanie zmiennych po angielsku;**

Typy danych

```
let age = 20;

let name = “Adam”;

let isTrue = true;

let variable;

let test = null;
```

Number - wartość liczbowa;

String - wartość znakowa;

Boolean - wartość logiczna (prawda lub fałsz);

Undefined - wartość domyślna;

Null - wartość bez wartości (typ wyświetli się jako objekt);

Łączenie wyświetlania tekstu i wartości zmiennej

```
const name = 'Adam';

let age = 20;

console.log(`Mam na imię ${name} i mam ${age} lat!`);
```

Możemy w łatwy sposób wyświetlać zarówno tekst jak i wartości przypisane do zmiennej jednocześnie przy pomocy pojedynczej funkcji.

</> OPERATORY

Operatory

+ dodawanie

- odejmowanie

* mnożenie

/ dzielenie

** potęgowanie

% modulo

++ inkrementacja

-- dekrementacja