



PODSTAWY PROGRAMOWANIA

Na przykładzie JavaScript

Paweł Kaczmarek

Co to jest program komputerowy

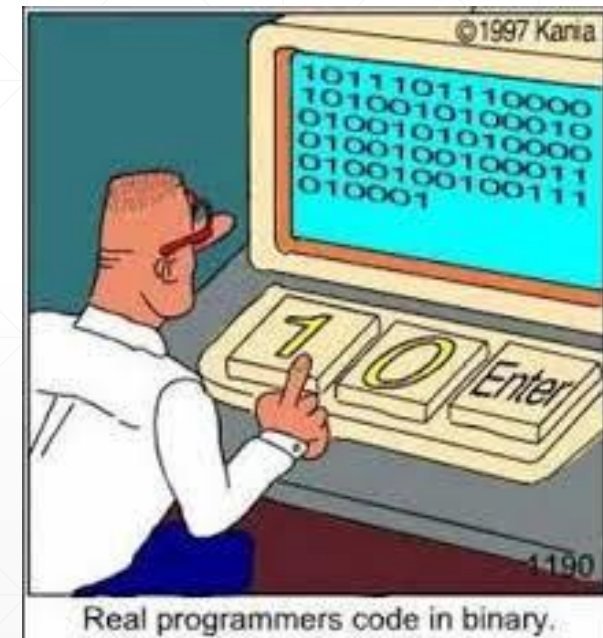
- Przeglądarka, Notatnik, Kalkulator, Spotify etc.

- `<script>`

```
    alert('Hello, World!');
```

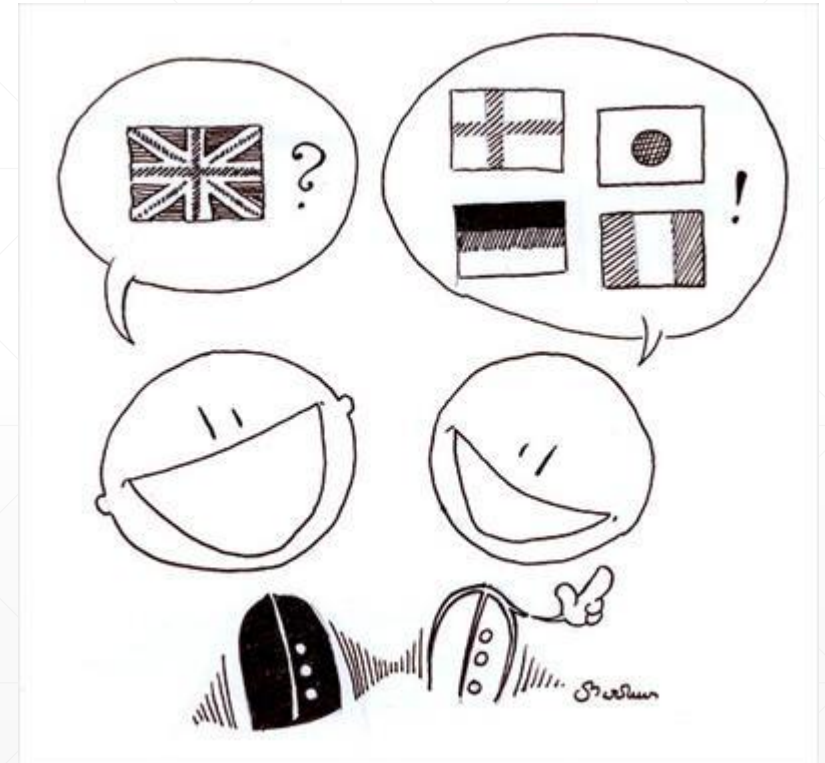
```
</script>
```

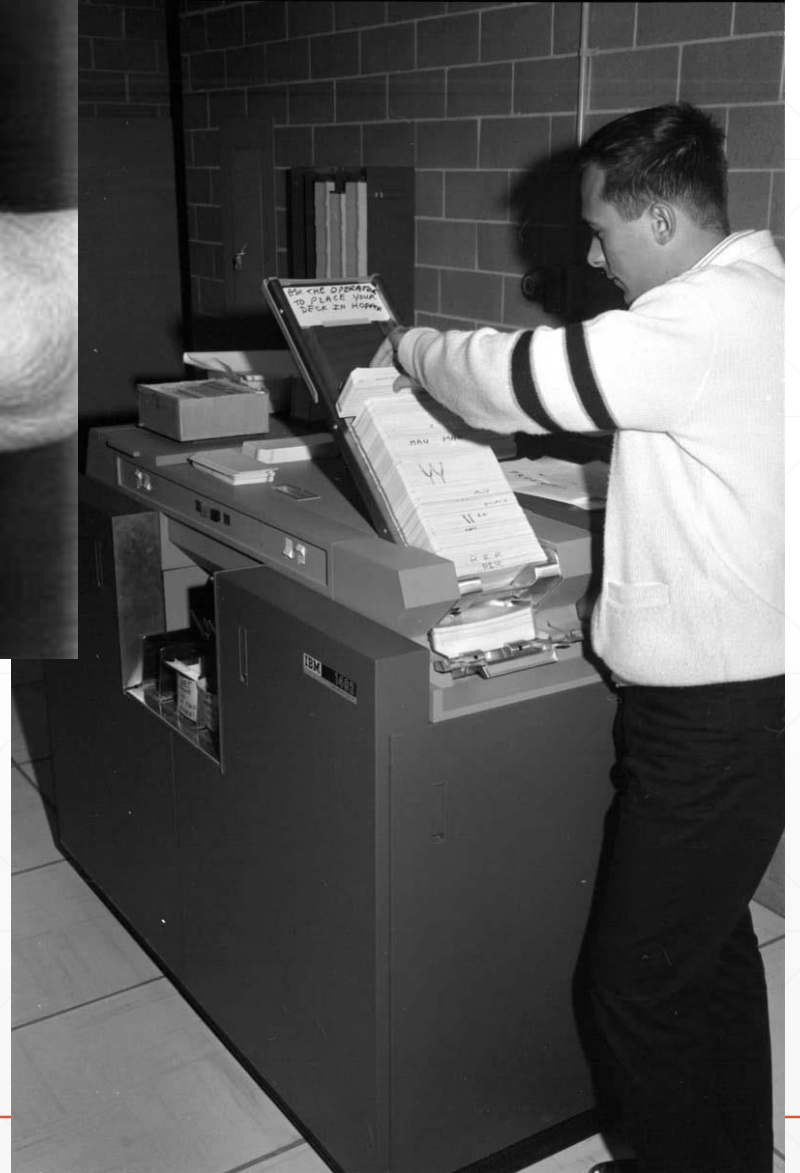
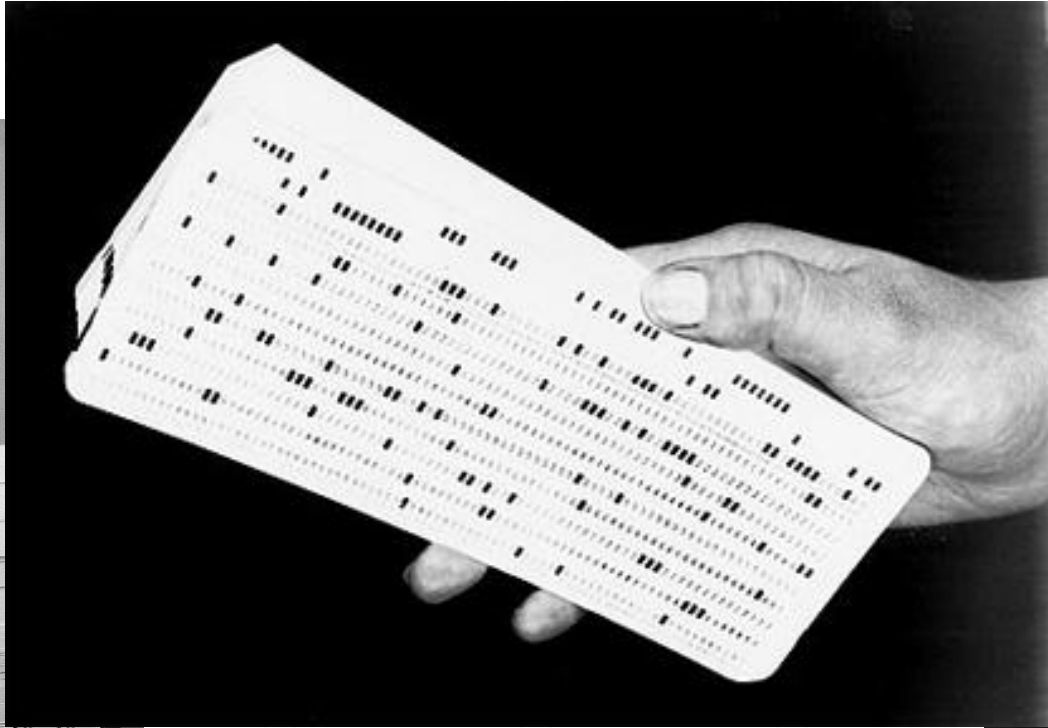
- 001011011110101011101001011010101...



Co to jest język programowania?

- Język obcy służący do komunikacji z komputerem
- Posiada składnię, gramatykę
- Wysokopoziomowe vs niskopoziomowe





Formatowanie textu

```
if (lang === 'pl')
|   {
|   showMsg('Witaj!');  }
|   else if (lang === 'de') {
|   showMsg ('Gutten Tag');
|   }
|   else {
|       showMsg('Hello');
|   }
```

```
var lang = 'pl';

if (lang === 'pl') {
|   showMsg('Witaj!');
|   }
|   else if (lang === 'de') {
|       showMsg ('Gutten Tag');
|   }
|   else {
|       showMsg('Hello');
|   }
```

Zmienne

Kontenery do przechowywania danych w programie

- Przechowują wartości – m.in. tekst, liczby
- Definiujemy je kiedy ich potrzebujemy

Zmienne – deklaracje, definicje

- **var** mojaWygrana ;
- **var** mojaWygrana = 100 ;
- **var** mojKomentarz = "Komentarz" ;
- **var** mojKomentarz = 'Komentarz' ;
- **var** mojaWygrana = 100 , mojKomentarz = 'Komentarz' ;

Liczba

- **var** zmienna = 100 ;
- **var** zmienna = 3.141592 ;

Tekst

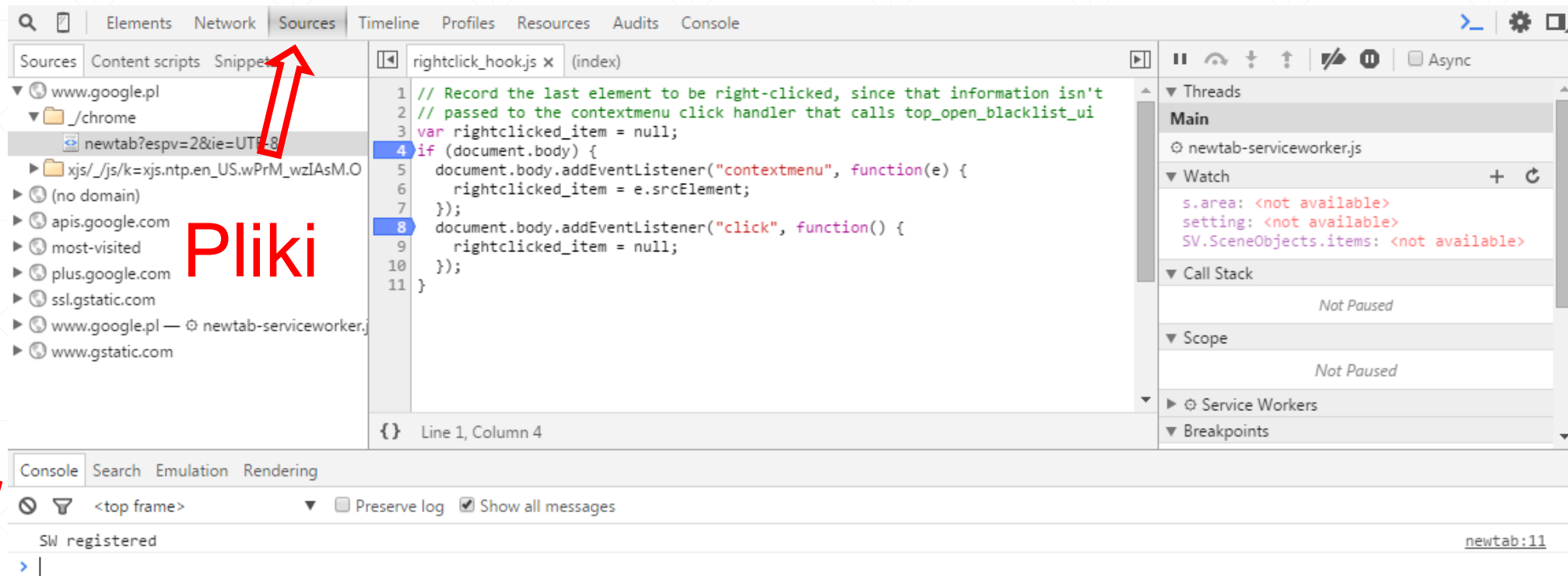
- **var** zmienna = 'jakiś text' ;
- **var** zmienna = '3.141592' ;

Operatory arytmetyczne

+	Dodawanie
-	Odejmowanie
*	Mnożenie
/	Dzielenie
++	Inkrementacja
--	Dekrementacja
%	Dzielenie modulo (reszta z dzielenia)

Operatory przypisania

=	$x = y$	$x = y$
+=	$x += y$	$x = x + y$
-=	$x -= y$	$x = x - y$
*=	$x *= y$	$x = x * y$
/=	$x /= y$	$x = x / y$
%=	$x \% = y$	$x = x \% y$



Konsola

Jak pisać na konsolę?

console.log ('tekst, który pokaże się na konsoli')

console.log (422)

Jak pisać na konsolę?

```
var mojaZmienna = ' moj Text ' ;  
console.log(mojaZmienna);
```

Komentarze (ctrl + /)

// pierwsza linia z komentarzem

```
console.log("linia 1");
```

// druga linia z komentarzem

```
console.log("linia 2");
```

*/**

*Wszystkie linie, aż do znaku
zamykającego komentarze
będą pominięte*

**/*

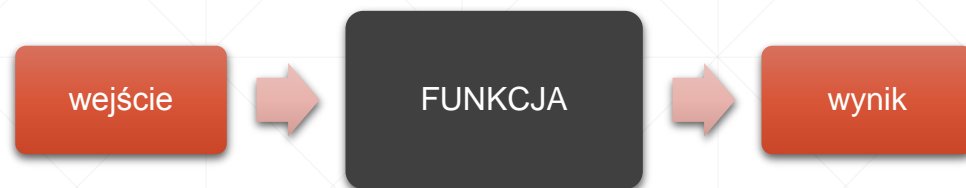
```
console.log("linia 1");
```

1a-zmienne



Funkcje





Funkcje

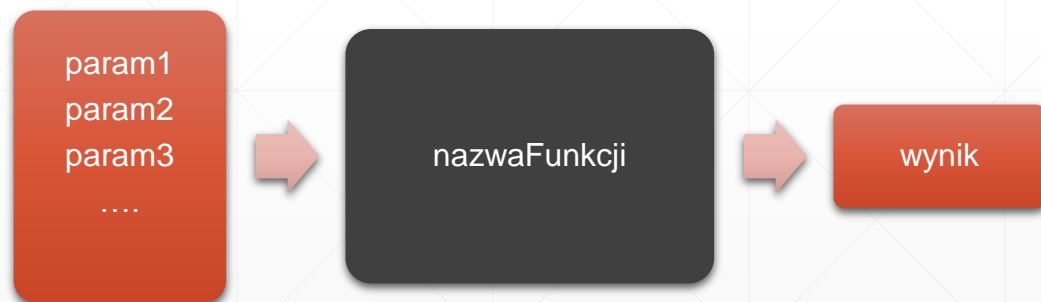
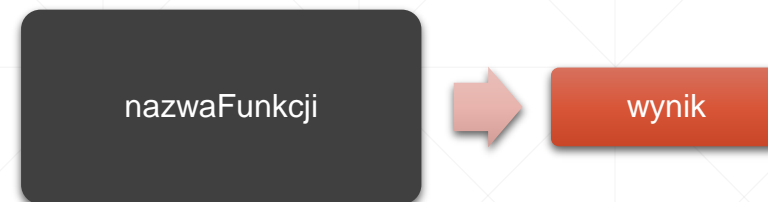
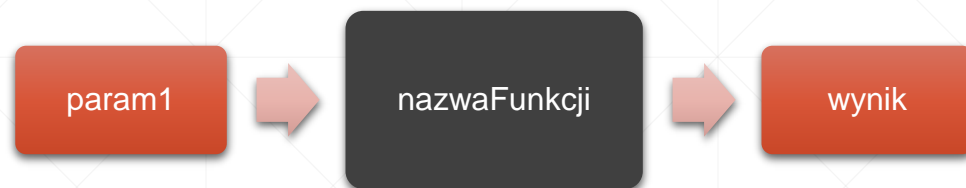
```
function nazwaFunkcji (parameter1, parameter2, parameter3) {
```

```
...
```

```
var wynik = parametr1 + parametr2 + parametr3 ;
```

```
return wynik;
```

```
}
```



```
function nazwaFunkcji (param1) {  
    return wynik;  
}
```

```
function nazwaFunkcji () {  
    return wynik;  
}
```

```
function nazwaFunkcji (param1, param2, param3) {  
    return wynik;  
}
```

```
function nazwaFunkcji (param1) {  
}
```

Wołanie funkcji bezparametrowej

```
function nazwaFunkcji () { ... }
```

```
nazwaFunkcji ( );
```

Wołanie funkcji z parametrami

```
function nazwaFunkcji (param1, param2) {  
    return wynik;  
}
```

```
nazwaFunkcji ( 100 , 'jakisText' );
```

Wołania funkcji

- `alert ('Komunikat')`
- `console.log('jakisText')`
- `fh2Celsius (120)`

```
function fh2Celsius (fahrenheit) {  
    return ( 5/9 ) * ( fahrenheit - 32 );  
}  
  
var celcius = fh2Celsius(100);  
console.log( celcius );
```

Definiowanie

vs

Wołanie

```
function nazwaFunkcji () {  
    ...  
}
```

```
nazwaFunkcji ( );
```

Definiowanie

vs

Wołanie

```
function nazwaFunkcji (p1, p2) {  
    ...  
}
```

```
nazwaFunkcji ( 10, 20 );
```


1b.funkcje



Zasięg zmiennych

GLOBALNE

```
var zmienna = 1;  
function policz() {  
    console.log( zmienna );  
}
```

LOKALNE

```
function policz() {  
    var zmienna = 1;  
}  
console.log( zmienna );
```

Instrukcje warunkowe

- if
- If-else
- switch

Operatory porównujące

- $A == B$
- $A != B$
- $A === B$
- $A !== B$
- $A < B$
- $A <= B$
- $A > B$
- $A >= B$

if

```
if ( warunek ) {
```

```
    // akcje zachodzące kiedy warunek jest prawdziwy
```

```
}
```

if

Jeżeli jest ładna pogoda to idziemy na spacer.

```
var ładnaPogoda;
```

```
if ( ładnaPogoda == true ) {
```

```
    idziemyNaSpacer();
```

```
}
```

if-else

Jeżeli jest ładna pogoda to idziemy na rower a jeżeli nie to pogramy na PS3

```
var ladnaPogoda;  
if ( ladnaPogoda == true ) {  
    idziemyNaSpacer();  
} else {  
    gramyNaPS3();  
}
```

2a.if



if-else-if

Jeżeli jest ładna pogoda to idziemy na spacer a jeżeli nie, to jak będziemy mieli dwa pady to pogramy na PS3. A jeżeli nie to obejrzymy jakiś film.

```
var ładnaPogoda, liczbaPadow;
```

```
if ( ładnaPogoda == true ) {  
    idziemyNaSpacer();  
} else if (liczbaPadow == 2) {  
    gramyNaPS3();  
} else {  
    ogladamyFilm();  
}
```

2b.if



switch

```
var mojaZmienna;
```

```
switch ( mojaZmienna ) {  
    case wartosc1 :  
        ....  
        break;  
    case wartosc2 :  
        ...  
        break;  
    case wartosc3 :  
        ...  
        break;  
    default:  
        ...  
}
```

switch

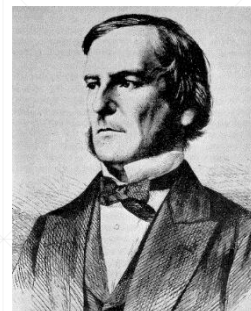
Jeżeli trafię 6tkę to pojadę dookoła świata, jeżeli 5tkę to kupię komputer, jeżeli 4kę to znowu zagram. W każdym innym przypadku dam sobie spokój.

```
switch ( liczbaTrafionychLiczba ) {  
    case 6:  
        jedziemyNaRocznyUrlop();  
        break;  
    case 5:  
        kupujemyKomputer();  
        break;  
    case 4:  
        gramyDalej();  
        break;  
    default:  
        odpuszczamySobie();  
}
```

2c.switch



Działania logiczne



AND

$$X = A \cdot B$$

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

OR

$$X = A + B$$

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

NOT

$$X = \bar{A}$$

A	X
0	1
1	0

Operatory logiczne

- **||** lub *suma logiczna*
- **&&** i *iloczyn logiczny*
- **!** negacja

```
var x, y, z;
```

```
if ( x ) { ... }
```

```
if ( ! x ) { ... }
```

```
if ( x || y ) { ... }
```

```
if ( x || y || z ) { ... }
```

```
if ( x || y && z ) { ... }
```

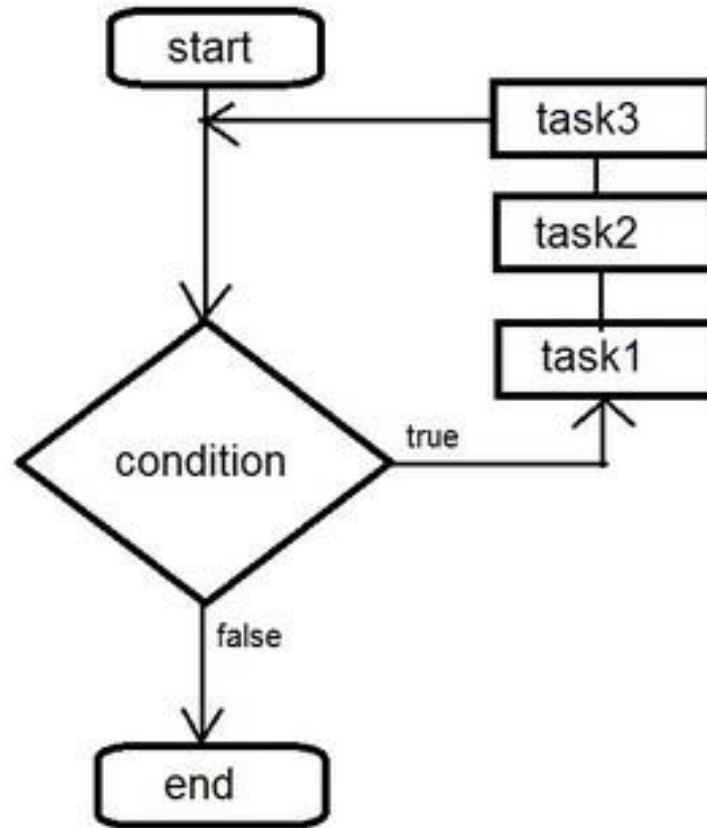

2d.complexIf



Petle

- while
- do-while
- for
- break
- continue

Pętle



while

Zagram w ruletkę 10 razy. Po czym wstanę i odejdę.

```
while ( condition ) {  
    ...  
}
```

```
var games = 0;  
while ( games < 10 ) {  
    playAgain();  
    games = games + 1;  
}
```

3a.while



do-while

```
doWork();
```

```
while ( condition ) {
```

```
    doWork();
```

```
}
```

do-while

```
doWork();
```

```
while ( condition ) {
```

```
    doWork();
```

```
}
```

===

```
do {
```

```
    doWork();
```

```
}
```

```
while ( condition )
```

do-while



Gwarantuje przynajmniej jedno wykonanie pętli.

Zagram w ruletkę. Będę grał tak długo dopóki nie będę miał mniej niż 100 zł.

```
do {  
    ...  
}  
while ( condition )
```

```
var winnings;  
  
do {  
    playAgain();  
}  
while ( winnings < 100 )
```


for

Wykonuje określoną liczbę powtórzeń

```
for ( inicjalizacja ; warunek ; zmiana ) {  
    ...  
}
```

```
for ( var i = 0 ; i < 6 ; i = i + 1 ) {  
    console.log( i );  
}
```

```
for ( inicjalizacja ; warunek ; zmiana ) {
```

```
    ...
```

```
}
```

```
for ( ; warunek ; zmiana ) {
```

```
    ...
```

```
}
```

```
for ( ; warunek ; ) {
```

```
    ...
```

```
}
```

3b.for



break

Wyjście z pętli



```
while ( .... ) {  
    ...  
    ...  
    if ( ... ) {  
        break;  
    }  
}
```

```
do {  
    ...  
    ...  
    if ( ... ) {  
        break;  
    }  
} while ( ... )
```

```
for ( ... ; ... ; ... ) {  
    ...  
    ...  
    if ( ... ) {  
        break;  
    }  
}
```

continue



Zakończenie bieżącego cyklu pętli

```
while ( .... ) {  
    ...  
    ...  
    if ( ... ) {  
        continue;  
    }  
}
```

```
do {  
    ...  
    ...  
    if ( ... ) {  
        continue;  
    }  
} while ( ... )
```

```
for ( ... ; ... ; ... ) {  
    ...  
    ...  
    if ( ... ) {  
        continue;  
    }  
}
```



PODSTAWY PROGRAMOWANIA

Dziękuję za uwagę!

Nice to haves

- condition ? dolfTrue() : dolfFalse()