# 1. Wprowadzenie do JS

**JavaScript** - język skryptowy, obiektowy rozszerzający standardowy HTML m.in. o możliwość interakcji z użytkownikiem przeglądającym stronę. Skrypty służą najczęściej do zapewnienia interaktywności poprzez reagowanie na zdarzenia (np. kliknięcie lub najechanie myszki), sprawdzania poprawności formularzy lub budowania elementów nawigacyjnych.

**Zastosowania:**

* Zdarzenia (reakcja na kliknięcie lub ruch myszy, dynamiczna zmiana zawartości i stylu)
* Animacje na stronie (np. galeria LightBox, proste gry, animowany zegarek)
* Przetwarzanie formularzy po stronie klienta
* Kalkulatory, testy on-line
* Sprawdzanie poprawności wprowadzanych danych do formularzy (walidacja) przed wysłaniem do serwera
* AJAX (przeładowanie – interakcja z użytkownikiem bez przeładowywania całej strony)
* Ciasteczka (zapamiętanie preferencji użytkownika – np. kolor tła strony, ułożenie menu itp.) .

**Cechy:**

* Język interpretowany po stronie klienta (przez przeglądarkę internetową)
* Kod programu jest umieszczony w kodzie HTML lub dołączany z zewnętrznych plików
* Język posiada predefiniowane obiekty (window, history, document, location, Array, Date, Math)
* Zmienne i ich typ nie muszą być deklarowane przed użyciem
* JavaScript rozróżnia wielkość liter
* Ze względów bezpieczeństwa nie ma możliwości zapisu na dysk twardy

# 2. Umieszczanie skryptu w dokumencie HTML

Umieszczanie skryptu w dokumencie html realizuje się poprzez znacznik <script></script>

c

**Przykład:** Wstawienie skryptu z zewnętrznego pliku *funkcje.js*

<script src="funkcje.js" type="text/javascript"></script>

**Przykład:** Wyskakujące okno dialogowe

<script type="text/javascript">

alert("Mój pierwszy skrypt");

</script>

# 3. Zmienne

**Zmienne** służą do przechowywania tymczasowych wyników obliczeń. Zmienne mają trzy podstawowe atrybuty: symboliczną nazwę, typ i wartość. Nazwa służy do identyfikowania zmiennej w związku z tym często nazywana jest identyfikatorem. Miejsce przechowywania przeważnie znajduje się w pamięci komputera. Liczba bajtów, potrzebnych do przechowania zmiennej w pamięci zależy od typu zmiennej. Zmienna jak sama nazwa wskazuje zmienia się podczas działania programu. Np. zmienna sekunda wykorzystywana w zegarku zmienia swoją wartość co sekundę według algorytmu odmierzania czasu.

**Z**mienne w JS deklarujemy za pomocą słowa kluczowego ***var*** np.

var x;

var y=5, z=6;

**Uwaga:** W języku Java Script nie musimy deklarować typu zmiennej, a jej typ określa przechowywana w niej wartość. W językach takich jak C lub Java należy deklarować typ zmiennej przed ich użyciem podając typ zmiennej i nazwę.

Podstawowe typy zmiennych w JavaScript:

**Zmienne liczbowe:** liczby całkowite, rzeczywiste

var x=5.5;

**Zmienne logiczne:** posiadają dwie wartości true i false

var w=false;

**Zmienne tekstowe:** znaki,łańcuchy znaków

var tekst = ”Ala ma kota”;

**Zmienne tablicowe:** przechowują wiele danych identyfikowanych za pomocą indeksu liczbowego lub tekstowego

var cars = ["Saab", "Volvo", "BMW"];

document.write( cars[0] ); // wyswietli pierwszy element tablicy

**Uwaga:** jeśli chcemy w zmiennej tekstowej dodać znak specjalny np. cudzysłów (”) wtedy stosujemy konstrukcję \”. \t – tabulator, \n – znak końca linii

Identyfikatory (nazwy) zmiennych podlegają pewnym ograniczeniom:

* Nazwy musza zaczynać się od litery bądź znaku podkreślenia (nie mogą zaczynać się np. od cyfr)
* Nazwy nie mogą zawierać znaków specjalnych i polskich liter (np. operatorów +/-\*% itp.)\
* Jako nazwy zmiennych nie wolno stosować słów kluczowych języka (np. var, for, if)

**Zadanie 1:** Zadeklaruj dwie zmienne typu liczbowego pi oraz e oraz przypisz im przybliżone wartości.

# 4. Instrukcja document.write()

**Instrukcja document.write(argumenty);** pozwala w skrypcie wypisywać dane w stronie.

Przy stosowaniu instrukcji document.write() należy pamiętać, że aby wyświetlić liczbę, wystarczy ją po prostu umieścić bezpośrednio w nawiasie okrągłym, np.:

document.write(1.4);

Można tez wyświetlać wyniki operacji arytmetycznych, relacji czy logicznych:

document.write(3.4+7);

document.write(3>7);

Równie łatwo można wyświetlić zawartość zmiennej np. x:

document.write(x);

Jeśli jednak na ekranie ma się pojawić napis (ciąg znaków) nie będący wartością liczbową, musi on być ujęty w znaki cudzysłowu lub apostrofu, np.:

document.write("x=");

W jednej instrukcji document.write można także wyświetlić wiele oddzielnych wartości, zarówno liczbowych, zmiennych jak i łańcuchów znakowych, należy wtedy użyć między nimi operatora +, np.:

document.write("x=" + x + "<br/>");

Istnieje także możliwość formatowania tekstu wewnątrz instrukcji:

document.write("<b>x=</b>" + x + "<br/>");

# 5. Operatory

Operatory są to symbole, które służą do wykonywania operacji na zmiennych. Przykład wyrażenia: c=a+b; do zmiennej c zostanie przypisana (=) wartość sumy (+) zmiennych a i b.

**Operatory przypisania** - służące do przypisywania zmiennym wartości, wynik wyrażenia z prawej strony przypisujemy do zmiennej znajdującej się z lewej strony.

|  |  |  |  |
| --- | --- | --- | --- |
| **Operatory przypisania** | | | |
| **operator** | **działanie** | **przykłady** | **wynik** |
| = | przypisanie | x=8  y=x | x ← 8  y ← 8 |
| += | przypisanie z dodawaniem | x=8  x+=3 | x ← 8  x ← 11 |

**Wskazówka:** Skrócone operatory przypisania: Zapis x=x+3 jest równoważny skróconemu zapisowi x+=3. Podobną konstrukcję można zastosować do innych operatorów np. /=, \*=, %= -=

**Operatory arytmetyczne –** służą do wykonywania operacji matematycznych

|  |  |  |  |
| --- | --- | --- | --- |
| **Operatory arytmetyczne** | | | |
| **operator** | **działanie** | **przykłady** | **wynik** |
| + | dodawanie | 5+7 | 12 |
| - | odejmowanie | 5-7 | -2 |
| \* | mnożenie | 5\*7 | 35 |
| / | dzielenie | 5/7 | 0.71428571428 |
| % | Reszta z dzielenia (modulo) | 10%4 | 2 |
| ++ | **inkrementacja**  zwiekszenie o 1 | x=4  x++ | x ← 4  x ← 5 |
| -- | **dekrementacja**  zmniejszenie o 1 | x=4  x-- | x ← 4  x ← 3 |

**Operatory porównania i relacyjne** – niezbędne w instrukcjach warunkowych, wynikiem wyrażenia porównania lub relacji jest prawda lub fałsz (true lub false)

|  |  |  |  |
| --- | --- | --- | --- |
| **Operatory porównania i relacji** | | | |
| **operator** | **działanie** | **przykłady** | **wynik** |
| == | równy | 5==7 | false |
| != | różny | 5!=7 | true |
| > | większy | 5>7 | false |
| >= | większy lub równy | 5>=5 | true |
| < | mniejszy | 5<7 | true |
| <= | większy lub równy | 5<=3 | false |

**Operatory logiczne** – wynikiem operacji logicznej jest true lub false. Operatory te służą do konstruowania złożonych wyrażeń logicznych.

|  |  |  |  |
| --- | --- | --- | --- |
| **Operatory logiczne** | | | |
| **operator** | **działanie** | **przykłady** | **wynik** |
| && | Iloczyn logiczny (AND)  “I” | false && false  false && true  true && false  **true && true** | false  false  false  **true** |
| II | suma logiczne (OR)  “LUB” | **false && false**  false && true  true && false  true && true | **false**  true  true  true |
| ! | negacja logiczna (NOT)  “NIE” | !false  !true | true  false |

**Wskazówka:** Operator AND daje prawdę tylko wtedy, gdy wszystkie argumenty mają wartość true. Operator OR daje fałsz tylko wtedy, gdy wszystkie argumenty mają wartość false.

**Operatrory bitowe** – wykonują operacje na poszczególnych bitach liczb. Przymiotnik bitowe oznacza, że mamy do czynienia z pojedynczymi bitami, lub liczbami binarnymi.

|  |  |  |  |
| --- | --- | --- | --- |
| **Operatory bitowe** | | | |
| **operator** | **działanie** | **przykłady** | **wynik** |
| & | iloczyn bitowy (AND)  bit=1 wtedy, gdy oba bity na danej pozycji są 1 | 6 & 5  6 = 110 (2)  5 = 101 (2)  100 (2) | 4 |
| I | suma bitowa (OR)  bit=1 wtedy, gdy jakikolwiek bit na danej pozycji jest 1 | 6 | 5  6 = 110 (2)  5 = 101 (2)  111 (2) | 7 |
| ^ | różnica symetryczna (XOR)  bit=1 wtedy, gdy bity na danej pozycji są różne | 6 ^ 5  6 = 110 (2)  5 = 101 (2)  011 (2) | 3 |
| ~ | negacja bitowa (NOT)  powoduje zmianę stanu bitu na przeciwny | ~1  1 = 001 (2)  ~1 = 110 (2) | 6  Uwaga – w zapisie 3 bitowym |
| >> | przesunięcie bitowe w prawo | 6>>1  6 = 110 (2)  011 (2) | 3 |
| << | przesunięcie bitowe w lewo | 2<<1  2 = 010 (2)  100 (2) | 4 |

**Zadania**

**Zadanie 1.** Uzupełnij tabelę wpisując odpowiednie wyniki działania operatorów

**Założenie:** x = 11, y=5

|  |  |  |
| --- | --- | --- |
| **Działanie** | **Wynik działania** | **typ operatora** |
| x+y |  | arytmetyczny |
| y-x |  |
| x%y |  |
| y++ |  |
| x>y |  |
| y<=10 |  | relacji |
| !(x>10) |  |
| x!=y |  |
| (x<5) && (y<5) |  | relacji, logiczny |
| (x<5) || (y<5) |  |
| x & 5 | 1011 = 11  & 0101 = 5  -----------------  0001 = 1  **wynik** | bitowy |
| x ^ y |  |
| x |y |  |
| ”x=”+x |  | konkatenacji, arytmetyczny |
| ”x\*y=” + (x\*y) |  |

**Zadanie 2.** : Jaki jest wynik działania poniższego programu

var x = 6;

var y = 5;

document.write( x!=y );

**Zadanie 3.** : Jaki jest wynik działania operacji: (5<7) && (5!=7)

**Zadanie 4:** Napisz program, w którym zadeklarujesz trzy zmienne liczbowe (a,b,c), przypiszesz im dowolne wartości oraz obliczysz i wyświetlisz ich średnią.

**Zadanie 5:** Wyświetl zawartość zmiennej x w postaci ciągu **Liczba x wynosi 5**

var x=5;

document.write( );

# 6. Okna dialogowe

|  |  |  |
| --- | --- | --- |
| **Nazwa funkcji** | **Przykład wywołania** | **Działanie programu** |
| **alert ( tekst )** - funkcja wywołuje okno powiadomienia wyświetlające tekst i przycisk OK, którym użytkownik zamyka okno. | alert(”Komunikat !!!”) |  |
| **prompt ( tekst, wartośćDomyślna )** - funkcja wywołuje okno wczytywania danych wyświetlające tekst oraz pole tekstowe o wartości domyślnej wartośćDomyślna (drugi parametr nie jest wymagany). Rezultatem funkcji jest wprowadzony tekst, po naciśnięciu przycisku OK lub wartość null, po naciśnięciu przycisku Cancel (Anuluj). | **var** x = prompt("Wprowadź liczbę");  document.write(x\*x); | 4 |
| **confirm ( tekst )** – funkcja wywołuje okno powiadomienia wyświetlające tekst oraz przyciski OK i Cancel (Anuluj). Rezultatem jest true gdy naciśnięto OK i false gdy naciśnięto Cancel (Anuluj). | **var** f = confirm(”Chcesz wejść?”);  **if**(f) {  document.write(”OK”);  } **else** {  document.write(”Anulowałeś”);  } | OK |

**Zadania**

**Zadanie 1:** Napisz program, który po wprowadzeniu w oknie dialogowym swojego imienia (np. Jan) wypisze komunikat: **Witaj Jan**

**Zadanie 2:** Napisz program, który po wprowadzeniu w oknach dialogowych zmiennych x i y wyświetli ich iloczyn

**Zadanie 3:** Napisz program, który po wprowadzeniu w oknach dialogowych zmiennych x i y wyświetli w kolejnych wierszach wynik działania operacji arytmetycznych (+, -, \*, /) w postaci

x=4

y=6

x+y=10

x-y=-2

x\*y=24

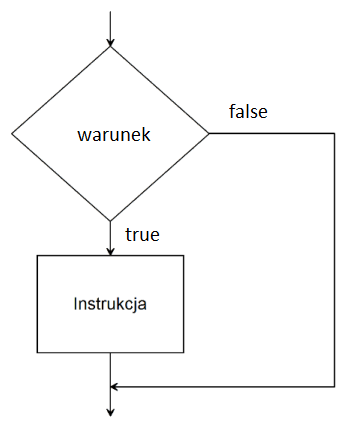
x/y=-0.66666666666

**Uwaga:** Funkcja prompt() zwraca wynik w postaci zmiennej tekstowej i operacja + spowoduje połączenie łańcuchów znakowych do postaci 46. Żeby tego uniknąć należy przekonwertować zmienne do typu liczbowego np. za pomocą operacji x=parseFloat(x);

var x=prompt(”Podaj x”);

x= parseFloat(x);

# 7.Instrukcje warunkowe

Instrukcje warunkowe pozwalają wykonać określony fragment kodu w zależności od spełnienia jakiegoś warunku. Służy do tego konstrukcja instrukcji warunkowej. W najprostszej wersji fragment programu będzie wykonany jeśli warunek zostanie spełniony:

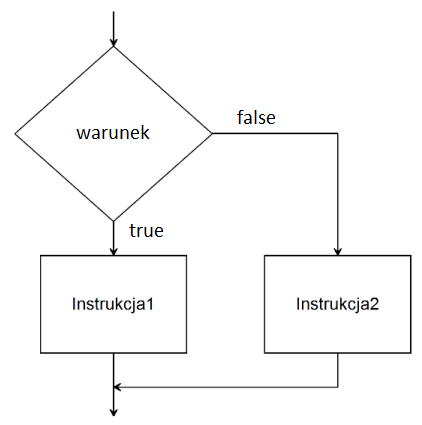
**if**(warunek)

{

instrukcja;

}

Druga postać instrukcji warunkowej umożliwia wykonanie pierwszego bloku instrukcji, gdy warunek zostanie spełniony i drugiego w przeciwnym przypadku:



**if**(warunek)

{

instrukcja1;

}

**else**

{

instrukcja2;

}

**Przykład 1.** Program wyznaczający wartość bezwzględną liczby x

**if**(x<0) { x=-x; }

**Przykład 2.** Program wyświetlający większą z dwóch liczb

**var** a=prompt(”Wprowadź pierwszą liczbę”);

**var** b=prompt(”Wprowadź drugą liczbę”);

**if**(a>b)

{

document.write(”Większa liczba to: ” + a);

}

**else**

{

document.write(”Większa liczba to: ” + b);

}

**Przykład 3.** Program, który na podstawie danej liczby wyświetla jeden z trzech komunikatów: Liczba jest mniejsza niż 0, Liczba jest większa niż 0, Liczba jest równa 0

**var**x=prompt(”Wprowadź liczbę”);

**if**(x<0)

{

document.write(”liczba jest mniejsza niż 0”);

}

**else if**(x>0)

{

document.write(”liczba jest większa niż 0”);

}

**else**

{

document.write(”liczba jest równa 0”);

}

**Zadania**

**Zadanie 1:**Napisz program sprawdzający, czy osoba zapytana przez program o wiek jest pełnoletnia i wyświetlający komunikat ”Jesteś pełnoletni/a”.

Zadanie 2. Napisz program sprawdzający czy liczba, wprowadzona z klawiatury jest parzysta. Wykorzystaj fakt, że liczba jest parzysta jeśli reszta z dzielenia przez 2 jest równa 0 np. a%2==0.

Zdanie 3. Wykorzystując warunek istnienia trójkąta mówiący, że dla danych trzech odcinków o długości a, b, c można zbudować trójkąt, jeżeli: (a<b+c) oraz (b<a+c) oraz (c<a+b) napisz program, który na podstawie trzech wprowadzonych liczb wyświetli komunikat „Można zbudować trójkąt” albo „Nie można zbudować trójkąta” w zależności od spełnienia tego warunku.

# 8. Instrukcja wyboru switch-case

Instrukcja wyboru **switch** pozwala wykonać określony fragment kodu w zależności od wartości zmiennej. Zmienna jest porównywana z wartościami występującymi za słowem **case** i wykonywany jest fragmentu programu odpowiadający tej wartości. Po napotkaniu słowa kluczowego **break** następuje natychmiastowe opuszczenie instrukcji switch. W przypadku wartości nie występującej na żadnej pozycji case wykonana zostania instrukcja domyślna – **default:**

**Przykład:** Na podstawie wartości zmiennej ocena program wyświetla jej postać słowną

**var** ocena=4;

**switch**(ocena)

{

**case 1:** document.write(”Ocena niedostateczna”); **break;**

**case 2:** document.write(”Ocena dopuszczająca”); **break;**

**case 3:** document.write(”Ocena dostateczna”); **break;**

**case 4:** document.write(”Ocena dobra”); **break;**

**case 5:** document.write(”Ocena bardzo dobra”); **break;**

**case 6:** document.write(”Ocena celująca”); **break;**

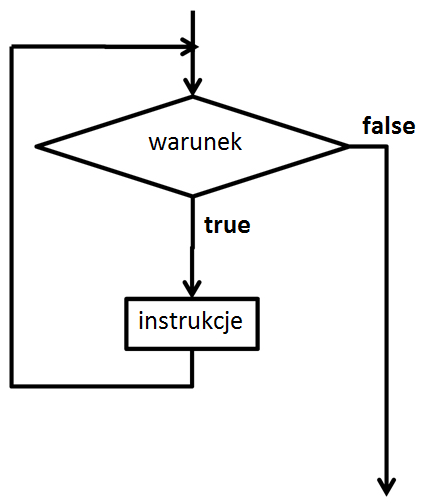
**default:** document.write(”Nie ma takiej oceny”); **break;**

}

# 9. Pętle programowe

Pętle programowe pozwalają na wielokrotne wykonanie danego fragmentu kodu w zależności od spełnienia pewnego warunku. pewnego warunku.

**Iteracja** (łac. iteratio – powtarzanie) – czynność powtarzania (najczęściej wielokrotnego) tej samej instrukcji (albo wielu instrukcji) w pętli. Mianem iteracji określa się także operacje wykonywane wewnątrz takiej pętli.

W językach programowania występują trzy wersje pętli programowych:

Pętla ze sprawdzaniem warunku na początku

**while**(warunek)

{

instrukcje;

}

Pętla ze sprawdzaniem warunku na końcu

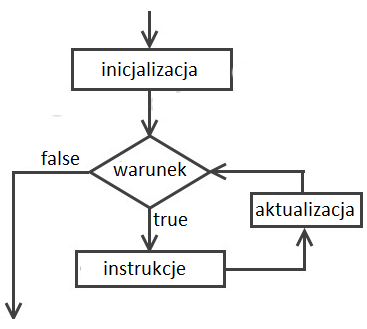
**do**

{

instrukcje;

}

**while**(warunek);

Pętla z licznikiem

**for**(inicjalizacja; warunek; aktualizacja)

{

instrukcje;

}

**Uwaga:** Użyciesłowa kluczowego **break** wewnątrz pętli powoduje natychmiastowe przerwanie wykonywania pętli. Słowo kluczowe **continue** pozwala na natychmiastowe przejście na początek pętli.

**Przykład 1.**Zastosowanie pętli ze sprawdzaniem warunku na początku do wyliczenia potęgi a^b

**var**a = 2;

**var**b = 4

**var** wynik = 1;

**while**(b>0)

{

wynik = wynik \* a;

b = b - 1;

}

document.write(wynik);

**Przykład 2.** Zastosowanie pętli ze sprawdzaniem warunku na końcu do sprawdzenia poprawności wprowadzenia danej liczbowej z zakresu 1..100.

**var** liczba;

**do**

{

liczba = prompt(”Podaj liczbę z zakresu 1..100”);

}

**while**( liczba<1 || liczba >100)

**Przykład 3.** Zastosowanie pętli z licznikiem do wyświetlenia liczb z zakresu 1..10

**for**(i=1; i<=10; i++)

{

document.write(i + ”, ”);

}

**Przykład 4.** Zastosowanie pętli z licznikiem do wyświetlenia liczb z zakresu 1..n i ich kwadratów

**var** n = prompt(”Podaj zakres”);

**for**(i=1; i<=n; i++)

{

document.write(i + ”^2 = ” + i\*i + ”<br>”);

}

**Zadania**

**Zadanie 1.** Wypisz w przeglądarce ciąg 1000 znaków np. #

**Zadanie 2.** Wypisz w przeglądarce ciąg liczb parzystych z zakresu 0..100

**Zadanie 3.** Wczytaj za pomocą funkcji prompt() dwie liczby a następnie wypisz ciąg o różnicy 1. Jeśli wprowadzisz dane 2 i 7 program powinien wypisać ciąg: 2, 3, 4, 5, 6, 7. W przypadku wprowadzenia liczb 6 i 1 program powinien wypisać ciąg: 6, 5, 4, 3, 2, 1. Zastosuj instrukcję warunkową do sprawdzenia rodzaju ciągu (rosnący, malejący).

# 10. Funkcje

Funkcja do fragment programu, który po zdefiniowaniu możemy wielokrotnie wywołać poprzez podanie jego nazwy i opcjonalnie argumentów. Stosowanie funkcji umożliwia tworzenie zwięzłych i czytelnych programów. Lista argumentów może być pusta, wtedy nie wpisujemy nic pomiędzy nawiasami okrągłymi (). W języku JavaScript istnieje wiele wbudowanych funkcji np. prompt(), alert(), document.write(), document.getElementById(), Math.sqrt(), itp. Istnieje też możliwość tworzenia i wywoływania własnych funkcji w programie. Jeśli funkcja ma pod swoja nazwą zwracać wartość wtedy stosujemy słowo kluczowe **return**.

**Przykłady funkcji:**

sin(x)

f(x) = x2 + 3x - 5

odleglosc(a,b) = √a2+b2

srednia(a,b,c) = (a+b+c)/3

**Definicja funkcji** określa jej nazwę, liczbę argumentów oraz definiuje jej działanie oraz zwracany wynik. **Wywołanie funkcji** przez program polega na podaniu jej nazwy z odpowiednimi argumentami (lub bez argumentów) umieszczionych pomiędzy nawiasami okrągłymi, co spowoduje wykonanie zawartego w niej kodu.

Struktura definicji funkcji:

**function** nazwa(argumenty)

{

instrukcje;

**return** wynik;

}

**Przykład 1.** Definicjafunkcji wyświetlającej linię znaków i jej trzykrotne wywołanie.

**function** linijka()

{

document.write(”\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*”);

}

linijka();

linijka();

linijka();

**Przykład 2.** Definicja funkcji obliczającej średnią dwóch liczb i jej wywołanie dla obliczenia średniej liczb 3 i 6.

**function** srednia(a,b)

{

return (a+b)/2;

}

**var** s = srednia(3,6);

document.write( s );

**Przykład 3.** Definicja funkcji obliczającej średnią trzech liczb na podstawie wprowadzonych przez użytkownika danych.

**function** srednia3(a,b,c)

{

return (a+b+c)/3;

}

**var** liczba1 = prompt(”podaj pierwszą liczbę”);

**var** liczba2 = prompt(”podaj drugą liczbę”);

**var** liczba3 = prompt(”podaj trzecią liczbę”);

document.write( srednia3(liczba1, liczba2, liczba3) );

**Uwaga.** Średnią 3 liczb można obliczyć też za pomocą funkcji dwuargumentowej z przykładu 2:

srednia(liczba1, srednia(liczba2, liczba3))

**Przykład 4.** Funkcja obliczająca największy wspólny dzielnik dwóch liczb (przydatna do skracania ułamków)

**function** nwd(a,b)

{

while(a!=b) {

if(a>b) a=a-b; else b=b-a;

}

**return** a;

}

// Wywołanie (użycie) zdefiniowanej wcześniej funkcji w programie

x = nwd(48,36);

document.write(x);

**Przykład 5:** Wykorzystanie wcześniej zdefiniowanej funkcji do definicji innej funkcji. Funkcja obliczająca najmniejszą wspólną wielokrotność (przydatna do obliczenia wspólnego mianownika).

**function** nww(a,b) { **return** a\*b/nwd(a,b); }

y = nww(48,36);

document.write(y);

**Podsumowanie**

* **nazwa** funkcji jest to dowolna nazwa, która powinna spełniać takie same wymogi jak nazwy zmiennych (czyli pierwszym znakiem może być litera lub znak podkreślenia; kolejne znaki nazwy mogą być literą, cyfrą lub znakiem podkreślenia; nazwa nie może też być zarezerwowanym słowem kluczowym).
* **argumenty** jest to lista nazw parametrów, rozdzielona przecinkami. Lista parametrów może być pusta (pomiędzy nawiasami okrągłymi wtedy nic nie ma).
* W ciele funkcji może być umieszczona dowolna liczba instrukcji i wywołań innych funkcji.
* Nawiasy klamrowe są obowiązkowe i nie można ich pominąć, nawet jeżeli funkcja zawiera tylko jedną instrukcję (lub nawet nie zawiera żadnej instrukcji).
* Funkcje wbudowane w język JavaScript nie wymagają definicji, można je od razu wywołać. Własne funkcje wywołuje się identycznie jak funkcje wbudowane - po prostu podaje się nazwę funkcji i w nawiasach okrągłych wartości parametrów.
* Funkcja może zwracać wartość – stosuje się wtedy słowo kluczowe **return.**

**Sprawdź swoją wiedzę**

**Pytanie: 1.** Zaznacz poprawny kod, który porównuje zmienną sum z wartością 65 i jeśli ich wartości są różne program wypisze "Sorry, try again"

a) if( 65 < sum ) alert("Sorry, try again" );

b) if( sum > 65 ) alert("Sorry, try again" );

c) if( 65 == sum ) alert("Sorry, try again" );

d) if( sum != 65 ) alert("Sorry, try again" );

**Pytanie 2.** Co wyświetli poniższy program jeśli zmienna x ma wartość 10

**if**(x>10)

{ alert(“Dużo”); }

**else**

{ alert(“Mało”) ; }

a) Dużo

b) Mało

c) Dużo Mało

d) Mało Dużo

**Pytanie 3.** Która składnia instrukcji switch jest poprawna?

a) switch(zmienna) { case 1: … break; case 2: … break; default: }

b) case(zmienna) { switch 1: … break; switch 2: … break; default: }

c) switch(zmienna) { case=1: … break; case=2: … break; default: }

d) case(zmienna) { switch=1: … break; switch=2: … break; default: }

**Pytanie 4.** Jakie słowo kluczowe powoduje natychmiastowe wyjście z instrukcji switch?

a) default

b) case

c) break

d) switch

**Pytanie 5.** Która wersja programu wyświetli na ekranie liczby 0,1,2,3,4,5,6

a) for(x=0; x<=6; x+=2) { document.write (x); }

b) for(x=0; x<=6; x++) { document.write (x); }

c) for(x=1; x<6; x+=2) { document.write (x); }

d) for(x=1; x<6; x++) { document.write (x); }

**Pytanie 6.** Ile razy poniższy program wyświetli napis OK!

i=0;

**while**(i<10) {

document.write(‘OK!’);

i+=3;

}

a) 2

b) 3

c) 4

d) 5

**Pytanie 7.** Jakiej instrukcji odpowiada schemat blokowy



a) **if**(warunek) { instrukcje; }

b) **while**(warunek) { instrukcje; }

c) **do** { instrukcje } **while**(warunek);

d) **for**(inicjalizacja; warunek; aktualizacja) { instrukcje; }

**Pytanie 8.** Która z poniższych definicji funkcji średnia jest poprawna

a) **function** srednia((a+b)/2) { **return**; }

b) **function** srednia(a, b) { (a+b)/ 2; }

c) **function** srednia(a; b) { **return** (a+b)/ 2; }

d) **function** srednia(a, b) { **return** (a+b)/ 2; }

**Pytanie 9.** Poprawne wywołanie funkcji o nazwie srednia z dwoma parametrami to

a) srednia(2 , 4);

b) srednia(2 ; 4);

c) srednia{2 , 4};

d) srednia{2 ; 4};

**Zadanie 1.** Napisz program, który na podstawie wprowadzonej liczby za pomocą funkcji prompt() wyświetli tyle razy napis ”JavaScript<hr>”

# 11. Zdarzenia

Zdarzenia w języku JavaScript to akcje, które mogą być inicjowane przez:

* zdarzenia myszki (np. kliknięcie, ruch myszy),
* zdarzenia klawiatury (np. wciśnięcie, zwolnienie klawisza) ,
* obsługę formularzy (np. wysłanie formularza),
* zdarzenia dokumentu (np. załadowanie dokumentu).

Programowanie zdarzeniowe oparte jest na wywoływaniu odpowiedniej funkcji w zależności od zaistnienia danego zdarzenia. DOM (*ang. Document Object Model*) dostarczają szereg zdarzeń, które wywoływane są kiedy zachodzi interakcja użytkownika ze stroną. Poniżej lista najważniejszych zdarzeń modelu DOM:

* **onAbort** - wywoływane jest w momencie zaniechania ładowania strony
* **onBlur** - wywoływany jest kiedy element przestaje być aktywny (traci “focus”)
* **obDblClick** - zdarzenie podwójnego kliknięcia w obiekt
* **onChange** - wywoływany jest w momencie gdy obiekt zmieni swoją zawartość
* **onClick** - zdarzenie kliknięcia elementu
* **onError** - wywoływany jest kiedy w skrypcie wystąpi błąd
* **onFocus** - wywoływany jest kiedy element staje sie aktywny (uzyskuje “focus”, przeciwieństwo ‘onBlur’)
* **onKeyDown** - wywoływany jest w momencie naciśnięcia klawisza klawiatury
* **onKeyUp** - wywoływany jest w momencie puszczenia klawisza klawiatury
* **onLoad** - wystepuje po załadowaniu elementu
* **onMouseOver** - występuje w momencie najechania na element kursorem myszki
* **onMouseOut** - występuje w momencie opuszczenia przez kursor myszki obiektu
* **onSelect** - wywoływany jest kiedy zawartość obiektu zostanie zaznaczona
* **onSubmit** - występuje w momencie zatwierdzenia formularza
* **onUnload** - wywoływany jest gdy strona zostanie zmieniona (np. kliknięto link i następuje przekierowanie)

**Przykład 1.** Najprostszy sposób polega na rejestracji zdarzenia w znaczniku HTML, który ma obsłużyć zdarzenia

<a href="index.html **onclick**="alert('Link do strony głównej')"> Strona główna </a>

**Przykład 2.** Wywołanie funkcji JavaScript o nazwie oblicz() dla przycisku formularza

<input type="button" value="Oblicz" **onclick="oblicz()"**>

**Przykład 3.** Zmiana koloru tła po najechaniu myszą na tekst

<script type="text/javascript">  
 function kolor\_tla(kolor)

{

document.bgColor = kolor;

}  
</script>

<p onMouseOver="kolor("red")">Czerwone tło</p>

<p onMouseOver="kolor("yellow")">Żółte tło</p>

# 12. Formularze

**Przykład 1.** Obsługa pól formularza w JavaScript. Kalkulator obliczający kwadrat i pierwiastek liczby

a) projekt formularza



1. kod formularza i skrypt

**<form>**

x=<input type="text" id="x" name="x">

<input type="button" value="kwadrat" **onclick="kwadrat()"**>

<input type="button" value="pierwiastek" **onclick="pierwiastek()"**>

**</form>**

<script type="text/javascript">

**function** kwadrat()

{

**var** x=document.getElementById('x').value;

alert(x\*x);

}

</script>

b) dopisz funkcję pierwiastek()

**Uwagi:**

* Dostęp do pól formularza identyfikowanego za pomocą atrybutów id="x": **document.getElementById('x').value;**
* Dostęp do pól formularza identyfikowanego za pomocą atrybutów name="x": **document.getElementsByName('x').value;**
* Funkcja **toFixed(2)** konwertuje liczbę zmiennoprzecinkową do dwóch miejsc po przecinku
* Aby zmienna tekstowa traktowana była jak liczba należy zastosować funkcję **parseFloat()** dla konwersji do typu zmiennoprzecinkowego lub **parseInt()** przy konwersji stałoprzecinkowej
* Funkcje matematyczne zgrupowane są w obiekcie Math np.: **Math.sqrt(a)** – pierwiastek z a, **Math.pow(a,b)** – potęga a^b

**Wyniki działania skryptu możemy wyświetlić na kilka sposobów:**

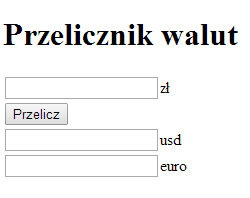
* za pomocą funkcji **alert()** – wynik pojawia się w wyskakującym okienku,
* za pomocą funkcji **document.write()** – wynik pojawia się na nowej stronie,
* poprzez zapis wyniku operacji do wnętrza bloku np. <div id="wynik"></div>

document.getElementById('wynik').innerHTML = x\*x;

* poprzez zapis wyniku operacji do innego pola formularza z identyfikatorem id="wynik"

document.getElementById('wynik').value = x\*x;

**Przykład 2.** Zaprojektuj formularz przelicznika walut oraz napisz funkcję realizującą konwersję. Wyniki działań powinny znaleźć się w odpowiednich polach formularza. Nazwij pola formularza odpowiednio zl, usd, euro.



**<form>**

<input type="text" id="zl">zł

<input type="button" value="Przelicz" **onclick="przelicz()"**>

<input type="text" id="usd">usd

<input type="text" id="euro">euro

**</form>**

<script type="text/javascript">

**function** przelicz()

{

**var** zl = document.getElementById('zl').value;

**var** usd = zl/4.2;

document.getElementById('usd').value = usd.toFixed(2);

}

</script>

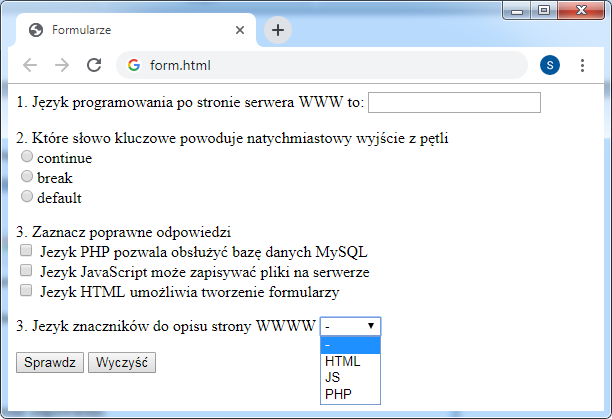
Uzupełnij funkcję o obsługę przelicznika dla euro

**Zadanie 1.** Zaprojektuj formularz prostego kalkulatora i napisz skrypt realizujący podstawowe operacje arytmetyczne



**Uwaga:** do każdego przycisku napisz oddzielną funkcję.

**Przykład 3 .** Zaprojektuj poniższy formularz. Każde pole opisz atrybutem name z wartościami p1, p2, p3, p4



<body>

<form>

<p>1. Język programowania po stronie serwera WWW to: <input type="text" name="p1"/></p>

<p>2. Które słowo kluczowe powoduje natychmiastowy wyjście z pętli<br/>

<input type="radio" name="p2">continue <br>

<input type="radio" name="p2">break <br>

<input type="radio" name="p2">default

</p>

<p>3. Zaznacz poprawne odpowiedzi

<br/>

<input type="checkbox" name="p3"> Jezyk PHP pozwala obsłużyć bazę danych MySQL<br>

<input type="checkbox" name="p3"> Jezyk JavaScript może zapisywać pliki na serwerze<br>

<input type="checkbox" name="p3"> Jezyk HTML umożliwia tworzenie formularzy<br>

</p>

<p>3. Jezyk znaczników do opisu strony WWWW

<select name="p4">

<option>-</option>

<option>HTML</option>

<option>JS</option>

<option>PHP</option>

</select>

</p>

<button type="button" onclick="sprawdz()">Sprawdz</button>

<button type="reset">Wyczyść</button>

</form>

<script type="text/javascript">

function sprawdz()

{

var punkty = 0;

var p;

p = document.getElementsByName("p1");

if(p[0].value="php") punkty++;

p = document.getElementsByName("p2");

if(p[1].checked) punkty++;

p = document.getElementsByName("p3");

if(p[0].checked && p[2].checked) punkty++;

p = document.getElementsByName("p4");

if(p[0].selectedIndex==1) punkty++;

alert("Zdobyte punkty: " + punkty);

}

</script>

</body>

Uwaga: Funkcja **document.getElementsByName()** zwraca wynik w postaci tablicy obiektów. Indeksy kolejnych pól liczymy od 0

# 13. Obiekty

Język JavaScript dostarcza wiele wbudowanych obiektów, z których możemy korzystać podczas tworzenia skryptów.

* [**Number**](http://www.w3schools.com/jsref/jsref_number.asp)- reprezentuje liczbę
* [**String**](http://www.w3schools.com/jsref/jsref_obj_string.asp)- umożliwia przechowywanie tekstu i manipulowanie nim
* [**Array**](http://www.w3schools.com/jsref/jsref_obj_array.asp) - służy do przechowywania tablic, czyli zmiennych zawierających wiele wartości
* [**Date**](http://www.w3schools.com/jsref/jsref_obj_date.asp) - używane jest do pracy z datami
* [**Math**](http://www.w3schools.com/jsref/jsref_obj_math.asp) - dostarcza funkcji matematycznych realizujących różnego rodzaju obliczenia
* [**RegExp**](http://www.w3schools.com/jsref/jsref_obj_regexp.asp) - umożliwia tworzenie wyrażeń regularnych

Każdy z tych obiektów posiada szereg funkcji (metod) pozwalających na pracę z danymi, które reprezentują. Metodę wywołuje się za pomocą operatora kropki "." .

var number = 10; // obiekt typu Number

var str = 'witaj swiecie'; // obiekt typu String

var tablica = new Array('Ala', 'Tomek', 'Krzysztof');

var data = new Date();

alert(str.length); // pobieranie dlugosci tekstu

alert(str.toUpperCase()) // konwersja na wielkie litery

alert(tablica.length); // wielkosc tablicy

alert(tablica[1]); // indeksu elementu

alert(d.getDate()); // pobieranie daty

alert(Math.PI); // wartosc liczby PI

alert(Math.sqrt(9)); //pierwiastek

alert(Math.round(1.14567)); // zaokrąglanie

Podobnie jak w innych językach programowania, wszystkie obiekty, zarówno natywne jak i własne, dziedziczą z obiektu ‘Object’. Tworzenie obiektów odbywa się poprzez użycie słowa kluczowego ‘new’. W JavaScript nie ma jednak definicji klasy. Zamiast tego pisze się funkcję, którą później wywoła się jak konstruktor, właśnie za pomocą ‘new’. Atrybuty obiektu wskazuje się przez słowo kluczowe this.

function Samochod(nazwa) {

this.nazwa = nazwa;

this.kolor = "Red";

}

var opel = new Samochod('Astra');

alert(opel.nazwa);