Pętle

Rodzaje pętli

Stosujemy je gdy jakiś kod powinien być wykonany kilku/wielokrotnie,

dla skrócenia zapisu i czytelności kodu.

W JS występują pętle typowe dla języków programowania czyli:

- for
- while
- do … while
- foreach
- for … in

Różnią się one budową i zastosowaniem.

Pętla for

Stosuje się ją do wykonania pętli o określonej liczbie powtórzeń. Ma ona trzy argumenty:

- wyrażenie początkowe, wykonywane raz na samym początku
- wyrażenie warunkowe, sprawdzane każdorazowo przed kolejnym wejściem do pętli
- wyrażenie modyfikujące zmienną iteracyjną (sterującą pętlą) po wykonaniu instrukcji w pętli

oddzielone średnikami i instrukcję/blok instrukcji do wykonania.

Postać	Przykład
for (wyrażenie_Początkowe ; wyrażenie_Warunkowe ; wyrażenie_Modyfikujące) instrukcja;	`for (let i=1; i<=5; i++)` `  document.write(i+" ");`
for (wyr_początkowe ; wyr_warunkowe ; wyr_modyfikujące) { instrukcja1; instrukcja2; ….. }	`for (let i=1; i<=5; i++){` `  document.write(i);` `  document.write(" ");` `}`

Argumenty dla pętli *for*

Wyrażenie początkowe	Wyrażenie warunkowe	Wyrażenie modyfikujące
Może mieć postać: ➢ *let i=1;* ➢ *var i=1;* ➢ *i=1;* - o ile zmienna została wcześniej zadeklarowana w skrypcie Może też zostać pominięcie o ile wcześniej nastąpiło nadanie wartości zmiennej iteracyjnej (nie zaleca się takiego rozwiązania) i wygląda to wtedy jak w przykładzie: *let i=5;* *for (; i<10; i+=2){* *instrukcje;* *}*	Określa warunek, który musi być spełniony, aby pętla została wykonana kolejny raz, np.: ➢ *i<=100;* ➢ *j>=i;*	Modyfikuje wartość zmiennej iteracyjnej (sterującej pętlą), która jest licznikiem, np.: ➢ i*++;* ➢ *i+=3;* ➢ *i--;* ➢ *i*=2;* Może ono zostać pominięte, o ile modyfikacja następuje w pętli (nie zaleca się takiego rozwiązania) i wygląda to wtedy jak w przykładzie: *for (let i=1; i<=10;){* *instrukcja;* *i++;* *}*

Wyjście z pętli lub jej przejście

- Chociaż nie jest zalecane to jednak istnieje możliwość opuszczenia pętli za pomocą instrukcji *break;*.
- Można zwiększyć krok pętli w wyrażeniu modyfikującym, np. *i+=2;*.
- Można również zmienić wartość zmiennej iteracyjnej (sterującej pętlą) wewnątrz pętli. Takie zachowanie nie jest jednak istotą tej pętli.

Zagnieżdżanie pętli *for*

Można umieszczać pętlę w pętli. Należy jednak pilnować, aby zmienne sterujące pętlą się nie pokryły. Zmienne te zwyczajowo nazywamy i, j, k w kolejności jak podano.

Przykład	Realizacja
`for (let i=1; i<=5; i++){` `  for (let j=1; j<=6; j++)` `  document.write("* ");` `  document.write(" ");` `}`	* *

Pytania

Pytanie 1: Jak zadeklarować zmienną iteracyjną w pętli *for*?

Pytanie 2: Jak zatrzymać działanie pętli *for* przed zakończeniem wszystkich iteracji? Jak jeszcze można opuścić pętlę przed czasem?

Pytanie 3: Gdzie ustalamy krok modyfikacji dla pętli *for*?

Pytanie 4: Podaj postać pętli *for*.

Pytanie 5: Jakie nazwy najczęściej nadaje się zmiennym iteracyjnym (sterującym pętlą).

Zadania

Zadanie 1: Napisz skrypt, który zliczy sumę liczb od 101 do 200 bez korzystania ze wzoru Gaussa na sumę ciągu arytmetycznego.

Zadanie 2: Napisz skrypt, który po wczytaniu dwóch liczb całkowitych policzy sumę ich i wszystkich liczb znajdujących się między nimi (bez korzystania ze wzoru Gaussa. Skrypt powinien uwzględnić wszystkie możliwości takie jak pierwsza liczba jest większa, mniejsza lub równa drugiej.

Zadanie 3: Napisz skrypt, który po wczytaniu dwóch liczb całkowitych policzy sumę wszystkich liczb parzystych znajdujących się między nimi. Skrypt powinien uwzględnić wszystkie możliwości takie jak pierwsza liczba jest większa, mniejsza lub równa drugiej.

Zadania

Zadanie 4: Napisz skrypt, wyświetlający na ekranie trójkąt. Wysokość trójkąta wczytujemy z klawiatury. Poniższy trójkąt ma wysokość wynoszącą 5.

```
x
xx
xxx
xxxx
xxxxx
```

Zadanie 5: Napisz skrypt, wyświetlający na ekranie trójkąt. Wysokość trójkąta wczytujemy z klawiatury. Poniższy trójkąt ma wysokość wynoszącą 5.

```
    x
   xxx
  xxxxx
 xxxxxxx
xxxxxxxxx
```

Zadanie 3: Napisz skrypt, wyświetlający na ekranie tabliczkę mnożenia. Poniższa tabliczka jest do 5.

```
   | 1   2   3   4   5
=====================
1  | 1   2   3   4   5
2  | 2   4   6   8  10
3  | 3   6   9  12  15
4  | 4   8  12  14  20
5  | 5  10  15  20  25
```

Pętla *while*

Jeśli nie znamy ilości powtórzeń pętli, ale wiemy co warunkuje powtarzanie się pętli to dobrym rozwiązaniem jest użycie pętli *while*. Aby dało się wyjść z pętli należy wewnątrz umieścić instrukcję modyfikującą zmienną/zmienne wyrażenia warunkowego będącego argumentem pętli *while*.

Postać	Przykład

while (wyrażenie_Warunkowe)
instrukcja;
	`let i=1;`
	`while (i<10)`
	`  i++;`

while (wyrażenie_Warunkowe) {
instrukcja1;
instrukcja2;
…..
}
	`let i=1;`
	`while (i<10){`
	`  document.write(2*i);`
	`  i++;`
	`}`
	
	`let i=1, j=20;`
	`while (i<j){`
	`  document.write(i*j);`
	`  i++; j--;`
	`}`

Wykonanie pętli *while*

Pierwsze obliczenie wartości wyrażenia warunkowego odbywa się przed wykonaniem jakiejkolwiek instrukcji. Możliwa jest zatem sytuacja, gdy instrukcje w pętli czyli pętla nie zostanie wykonana ani razu.

Ponadto przed pętlą *while* należy dokonać przygotowania polegające na inicjalizacji zmiennych występujących w wyrażeniu warunkowym.

While jako pętla zaporowa

Zdarza się, że poprawność działania skrypu jest uwarunkowana poprawnością danych wejściowych, np. wprowadzonych przez użytkownika. Przykładowo jeśli użytkownik poda liczbę ujemną lub 0 jako długość boku figury, to powinien zostać poinformowany, że wymagana jest liczba dodatnia i poproszony o kolejną liczbę. Próbę wczytywania liczby powtarzamy dopóty, dopóki użytkownik nie poda liczby poprawnej (dodatniej). Taki sposób zapewnienia poprawności wczytywanych danych nazywać będziemy pętlą zaporową. Jako pętli zaporowej można użyć *while*.

```
let a=parseInt(prompt("Podaj bok kwadratu"));
while (a<=0){
    a=parseInt(prompt("Podaj bok kwadratu"));
}
```

Przerywanie i zagnieżdżanie pętli *while*

Podobnie jak w przypadku pętli *for* możliwe jest natychmiastowe opuszczenie pętli za pomocą instrukcji *break*.

Ponadto możliwe jest też zagnieżdżanie (umieszczanie pętli w pętli) przy czym dopuszczalne są tutaj różne układy, również z innymi pętlami:

- Pętla *while* wewnątrz pętli *while*
- Pętla *for* wewnątrz pętli *while*
- Pętla *while* wewnątrz pętli *for*

Jak przy każdym zagnieżdżaniu należy uważać, aby nie pomylić zmiennych iteracyjnych (sterujących pętlą) z różnych pętli.

Pytania

- Jak zapobiec tworzeniu pętli nieskończonej w pętli while?
- Jak zatrzymać działanie pętli while przed zakończeniem wszystkich iteracji?

Zadania

Zadanie 1: Wykorzystując pętlę zaporową napisz skrypt, wyznaczający pole kwadratu. Skrypt jako dane (długość boku) powinien przyjmować wyłącznie liczby dodatnie.

Zadanie 2: Napisz skrypt wyznaczający wartość pierwiastka kwadratowego z liczby. Wprowadzane dane zabezpiecz pętlą zaporową (użytkownik nie może wprowadzić liczby ujemnej bo wtedy wyjdzie NaN). Funkcja Math.sqrt(x) zwraca pierwiastek kwadratowy z podanego argumentu x.

Zadanie 3: Napisz skrypt, który wczytuje z klawiatury liczbę do momentu aż będzie ona liczbą dwucyfrową.

Zadanie 4: Napisz skrypt, który wczytuje z klawiatury poprawny numer miesiąca, tzn. liczbę z przedziału od 1 do 12. Zakładamy, że możliwe są tylko 3 próby podania poprawnego numeru.

Zadanie 5: Napisz skrypt, który dla wczytanej z klawiatury liczby n wypisuje na ekranie wszystkie liczby z przedziału 1...n podzielne przez 7 których ostatnia cyfra to 1.

Zadania cd.

Zadanie 6: Napisz skrypt, który wczytuje z klawiatury liczby rzeczywiste aż do momentu, gdy podana liczba jest równa zero, a następnie wyświetla na ekranie sumę i średnią arytmetyczną tych liczb.

Zadanie 7: Napisz skrypt, który drukuje na ekranie zadaną liczbę zastępując jej kolejne cyfry ich kwadratami (dla liczby 127 drukujemy 1449).

Zadanie 8*: Napisz skrypt, który dla wczytanej z klawiatury liczby naturalnej n wypisuje jej dzielniki. Zwróć uwagę do jakiej liczby trzeba przeszukiwać liczby będące potencjalnymi dzielnikami. https://www.algorytm.edu.pl/algorytmy-maturalne/algorytm-eulkidesa.html

Zadanie 9*: Napisz skrypt, który dla wczytanej z klawiatury liczby naturalnej n sprawdza, czy liczba ta jest pierwsza. https://www.algorytm.edu.pl/algorytmy-maturalne/badanie-czy-liczba-pierwsza.html

Zadanie 10*: Napisz skrypt, który dla wczytanych liczb naturalnych n, m oblicza ich NWW(n,m) https://www.algorytm.edu.pl/algorytmy-maturalne/nww.html

Pętla *do … while*

Pętla *do …while* zawiera instrukcje, które muszą się wykonać i powtarzać tak długo jak spełniony jest pewien warunek.

Pętli tej używamy zatem w sytuacji, gdy kod musi się wykonać przynajmniej jeden raz.

Postać	Przykład

do {
instrukcja1;
instrukcja2;
…..
} while (wyrażenie_Warunkowe) ;
	`let a;`
	`do {`
	`a=parseInt(prompt("Podaj bok kwadratu"));`
	`} while (a<=0);`

Pętla *do … while* – informacje dodatkowe

- Wewnątrz pętli musi znaleźć się instrukcja/instrukcje modyfikujące zmienne iteracyjne(sterujące pętlą)
- Deklaracja zmiennej iteracyjnej powinna się znaleźć przed pętlą
- Pętla kończy się średnikiem
- Awaryjnie z pętli można wyjść instrukcją *break*;
- Pętlę można zagnieżdżać

Pytanie: Podaj zasadniczą różnicę w zastosowaniu pętli *do … while* oraz *while*

Zadania

Zadanie 1: Napisz skrypt, który pobiera od użytkownika znak do momentu, gdy jest on literą 't', 'T', 'n', lub 'N'.

Zadanie 2:Napisz skrypt, który dla wczytanej z klawiatury liczby naturalnej wypisuje ilość cyfr tej liczby.

Zadanie 3: Napisz skrypt, który dla podanej liczby całkowitej nieujemnej n, obliczy sumę wszystkich jej cyfr i wyświetli otrzymany wynik.

Zadanie 4: Napisz skrypt, który obliczy ile 0 znajduje się w podanej liczbie.

Zadanie 5*: Napisz skrypt, który dla podanej liczby całkowitej nieujemnej n, obliczy sumę wszystkich jej cyfr i wyświetli otrzymany wynik, a następnie powtórzy te dwie czynności dla obliczonej sumy, itd. Ten proces winien być zakończony, gdy obliczona suma cyfr będzie liczbą jednocyfrową, np. dla n=7895, należy wyświetlić liczby: 29, 11, 2.

Zadanie 6*:Napisz skrypt, który podczas zamiany liczby podanej przez użytkownika w systemie dziesiętnym na system binarny znajduje liczbę jedynek (inaczej ile jest 1 w dwójkowym rozwinięciu podanej przez użytkownika liczby naturalnej n). Uwaga do zliczania 1 nie potrzebujemy tablicy. https://www.algorytm.edu.pl/funkcje/69-zamiana-liczby-dziesietnej-na-binarna.html