Uvod u programiranje

Funkcije

POWERED BY:

nsoft sportradar

Sadržaj

- funkcije
  - literals
  - pozivanje
  - parametri
  - return
  - varijable
  - ugnježđivanje
- rekurzivne funkcije

Funkcije

- grupa “reusable” koda koju pozivamo negdje u programu
- omogućuju programerima podjelu velikog programa u manje i praktične funkcije
- pomaže programerima za pisanja modularnog koda
  - alert(), prompt()

Funkcije

- koristimo ključnu riječ **function**, jedinstveno ime funkcije, lista parametara i izraz
- ime nije obavezno (**anonymous functions**)
- parametri nisu obavezni

```
function functionName (parameterList) {

   statements

}
```

Funkcije - literals

- izraz koji definira bezimenu funkciju

```
var functionName = function
(parameterList) {

   statements

}
```

Funkcije - pozivanje

```
function sayHello() {
   console.log('Hello');
}

sayHello();
```

```
<button onclick="sayHello()">Click me</button>
```

Funkcije - parametri

```
function sayHello(name) {
   console.log('Hello ' + name);
}

sayHello('Tim');
```

```
<button onclick="sayHello('Tim')">Click me</button>
```

Funkcije - return

- funkcija može imati **return** izraz
- potreban je ako želimo da funkcija vrati neku vrijednost

```
function sum(a, b) {
   return a + b;
}

console.log(sum(5, 6));
```

Funkcije - varijable (scope)

```
var x = 5;
console.log(x);

function change(num) {
   num = 10;
   console.log(num);
}

change(x);
console.log(x);
```

Funkcije - varijable (scope)

```
var x = 5;
console.log(x);

function change() {
  var x = 10;
  console.log(x);
}

change();
console.log(x);
```

Funkcije - varijable (scope)

```
var x = 5;
console.log(x);

function change() {
  x = 10;
  console.log(x);
}

change();
console.log(x);
```

Funkcije - varijable (closure)

- funkcija ima pristup varijablama definiranim izvan funkcije

```
var a = 5;

function sum() {
   return a + a;
}

sum(); // 10
```

Funkcije - ugnježđivanje

```
function hypotenuse(a, b) {
  function square(x) {
    return x * x;
  }

  return Math.sqrt(square(a) + square(b));
}

console.log(hypotenuse(3, 4));
```

Zadatak 1

Definirati funkciju *min()*, koja prima dva broja kao argumente i vraća manji od njih.

rješenje

Zadatak 2

Definirati funkciju *maxOfThree()*, koja prima tri broja kao argumente i vraća najveći od njih.

rješenje

Zadatak 3

Napisati funkciju koja će za proslijeđeni mjesec (brojčano), vratiti koliko ima dana u tom mjesecu.

Rezultat ispisati u glavnom programu.

rješenje

Zadatak 4

Napraviti funkciju za računanje broja na zadanu potenciju.

Funkcija prima dva parametra, broj i potenciju, te vraća rezultat.

U glavnom programu od korisnika zatražiti unos broja i potencije, te ispisati rezultat.

rješenje

Rekurzivne funkcije

- Rekurzija je proces u kojem funkcija poziva samu sebe
- Ako funkcija poziva samu sebe tada JS engine mora napraviti novi ‘stack’ (dio memorije alociran da bi cuvao trenutne informacije potrebne za izvršavanje funkcije)
- JS engine može napraviti ‘stack’-ova koliko ima dostupne memorije

Rekurzivne funkcije - primjer

Zbrajanje:

```
function sum(x, y) {
  return (y > 0) ? sum(x + 1, y - 1) : x;
}

console.log(sum(1, 6));
```

Rekurzivne funkcije - primjer

- proslijedimo dva broja (parametri **x** i **y**)
- provjeravamo je li **y** veće od **0**
- ako jeste, rekurzivno pozivamo **sum** ali ovaj put vrijednost argumenta **x** je uvećana za 1, a **y** umanjena za 1
- prilikom sljedećeg poziva funkciji prosljeđujemo **2** i **5**
- prvi poziv **sum** funkcije još nije završio
- u ovom trenutku JS engine ima 2 stack-a, jedan za x=1, y=6, a drugi za x=2, y=5
- JS mora napraviti stack za svaki rekurzivni poziv

Rekurzivne funkcije - primjer

Zbrajanje:

```
function sum(x, y) {
 return (y > 0) ? sum(x + 1, y - 1)
: x;
}

console.log(sum(1, 1000000));
```

→ zahtjeva puno više stack-ova nego što je dostupno memorije

Zadatak 5

Napraviti funkciju *countdown* koja prima jedan pozitivan broj i ispisuje sve brojeve do 0. (rekurzivno)

rješenje

Zadatak 6

Napraviti funkciju *factorial* koja računa faktorijele. (primjer: 5! -> 1*2*3*4*5)

rješenje

Zadatak 7

Napraviti funkciju *isEven* koja vraća **true** ako je broj paran, u suprotnom vraća **false**.

rješenje

POWERED BY: nsoft sportradar

SPARK. SCHOOL

Ponavljanje

POWERED BY: nsoft sportradar

SPARK. SCHOOL

THANK YOU
for attention