**Javascript 1 - Inlämningsuppgift 2 - Daniel Hessling**

**Del 1**

1.

2.

En funktion är ett block med kod som ska utföra någonting. T.ex en uträkning eller en text som ska skrivas ut. Fördelen med en funktion kan vara att man kan anropa den vart som helst och hur många gånger som helst i sin kod, På så sätt kan man slippa skriva om kod i onödan.

function myFunction(one, two)

{

Här skriver man vad man vill att funktionen ska utföra..

};

Man anropar denna funktion genom denna korta kodsnutt:

myFunction();

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Här är ett exempel:

function skrivNamn()

{

let namn;

namn = prompt("Ange ditt namn: ");

alert("Välkommen " + namn);

};

skrivNamn();

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ett annat exempel:

function Addition()

{

tal1 = prompt("Skriv ett tal: ");

tal2 = prompt("Skriv ett till tal: ");

resultat = +tal1 + +tal2;

alert("Summan av talen är: " + resultat);

};

Addition();

3.

En funktion kan ta emot värden genom att man skickar in parametrar inom parentesen efter funktionsnamnet, t.ex.

function minFunktion(**value1, value2**); Man kan skicka in redan deklarerade variabler som har ett förbestämt värde eller genom exempelvis en prompt();.

4. För att få en funktion att returnera ett värde så lägger man till en return som nedanstående exempel:

function myFunction()

{

let width = 30;

let height = 50;

let area = width \* height;

return area;

};

5.

Man kan returnera flera värden samtidigt genom att använda sig av en array eller ett objekt.

// Funktion som returnerar flera värden med en array

console.log(fleraVarden());

function fleraVarden() {

let treVarden = ["Värde 1","Värde 2","Värde 3"];

return treVarden;

6.

En anonym funktion är en funktion utan namn och som är lagrad i en variabel. Iomed detta så döper man inte funktionen till något. En till skillnad mot en vanlig funktion är att man inte kan anropa funktionen innan uttrycket.

7.

JavaScript är ett objekt-orienterat programmeringsspråk, nästan allting i JS är objekt. Man bygger ihop objekt som samspelar med varandra. Allt blir mer lätthanterligt och får en bra struktur.

8.

Man kan skapa objekt på olika sätt i JavaScript. En literal notation kan man t.ex använda när man bara vill spara data i ett objekt. De är bra att använda sig av KISS-principen (Keep it simple, stupid).

Constructor Notation använder man när man t.ex vill använda metoder i objektet. Constructor Notation låter dig skapa objekt med nya egenskaper.

”Literal notation” (objektliteral) och ”Constructor notation” (objektkonstruktor) är två olika sätt att skapa objekt på. Ska man inte lägga till flera egenskaper i objektet är literala metoden att föredra, vill man istället lägga till egenskaper så kan det vara lämpligt att istället använda konstruktor. Man bör tänka enligt KISS-principen, ”Keep It Simple, Stupid”.

// Objekt med två egenskaper som skapas med objektliteral

let literalObjekt = {

fornamn: "Andreas",

efternamn: "Åkerlöf"

}

// Objekt med två egenskaper som skapas med objektkonstruktor

let konstruktorObjekt = new Object();

konstruktorObjekt.fornamn = "Erika";

konstruktorObjekt.efternamn = "Åkerlöf";

9.

Egenskaper är information/data som finns inuti ett objekt. Man kan lägga till egenskaperna när man skapar ett objekt eller i efterhand.

let minInfo =

{

Namn: “Daniel”,

}

Behöver inte let

Egenskaper är data som finns i objekten. Egenskaper kan läggas till vid skapandet av objektet, eller i efterhand. Egenskaperna går att komma åt genom ”objektnamn.egenskap”.

// Objekt med en egenskap, får en till egenskap, sedan skrivs egenskaperna ut

let objektEgenskaper = {

namn: "Andreas"

}

objektEgenskaper.age = 34;

console.log(objektEgenskaper.namn + " är " + objektEgenskaper.age + " år.");

...

10.

Objekt kan ha funktioner i sig, dessa kallas för metoder. Metoder kan man skriva när man skapar ett objekt men det kan också läggas till i efterhand.

sen tar jag ett exempel som är lika som det förra fast med funktioner istället

Metoder kallas de funktioner som kan finnas i ett objekt. Metoder kan skrivas vid skapandet av ett objekt men kan också läggas till i efterhand precis som med egenskaper. När man anropar metoder skriver ”man objektnamn.metod()”.

// Objekt med en metod, som får en till metod, sedan anropas båda

let objektMetoder = {

namn: function() {

console.log("Andreas Åkerlöf");

}

}

objektMetoder.age = function() {

console.log(34);

}

objektMetoder.namn();

objektMetoder.age();

Ändra egenskaper........

**Del 2 - Uppgift 1**

Om man matar in nollor i programmet visar multiplikationen “0” medan divisionen visar “NaN”.

Om man istället matar in att tal 1 är > 0 och tal 2 = 0. Så visar multiplikationen “0” medan divisionen visar “Infinity”.

function Numbers()

{

let tal1 = prompt("Ange ett tal: ");

let tal2 = prompt("Ange ett tal till: ");

let multiply = tal1 \* tal2;

let divide = tal1 / tal2;

alert(tal1 + " \* " + tal2 + " = " + multiply);

alert(tal1 + " / " + tal2 + " = " + divide);

};

Numbers();

**Del 2 - Uppgift 2**

let width;

let height;

let area;

let circuit;

function Rectangle()

{

width = prompt("Ange bredden på rektangeln: ");

height = prompt("Ange höjden på rektangeln: ");

area = width \* height;

circuit = (width \* 2) + (height \* 2);

alert("Rektangelns area = " + area + "\n" + "Rektangelns omkrets = " + circuit);

}

Rectangle();

**Del 2 - Uppgift 3**

let fahrenheit;

let celsius;

function convertTemp()

{

fahrenheit = prompt("Ange temperaturen du vill omvandla från Celsius till Fahrenheit");

celsius = (5/9) \* (fahrenheit - 32);

alert(fahrenheit + " grader Fahrenheit = " + celsius + " grader Celsius");

}

convertTemp();

**Del 2 - Uppgift 4**

let Elevator =

{

floor: '',

goTo : function (goToFloor)

{

console.log("Going to floor: " + goToFloor)

floor = goToFloor;

},

where : function()

{

console.log("You are at floor: " + floor);

},

};

Elevator.goTo(9);

Elevator.where();