**Pràctica 11**

**Funcions**

En programació és molt freqüent que un determinat procediment de càlcul definit per un grup de sentències hagi de repetir-se diverses vegades, ja sigui en un mateix programa o en altres programes, la qual cosa implica que s'hagi d'escriure tants grups d'aquelles sentències com vegades aparegui aquest procés.

L'eina més potent amb què es compta per a facilitar, reduir i dividir el treball en programació, és escriure aquells grups de sentències una sola i única vegada sota la forma d'una FUNCIÓ.  
  
Un programa és una cosa complexa de realitzar i per tant és important que estigui ben ESTRUCTURAT i també que sigui intel·ligible per a les persones. Si un grup de sentències realitza una tasca ben definida, llavors pot estar justificat el aïllar aquestes sentències formant una funció, encara que resulti que només se li digui o faci servir una vegada.

Fins ara hem vist com resoldre un problema plantejant un únic algorisme.

Amb funcions podem segmentar un programa en diverses parts.

Davant d'un problema, plantegem un algoritme, aquest pot constar de petits algoritmes.

Una funció és un conjunt d'instruccions que resolen una part del problema i que pot ser utilitzat (anomenat) des de diferents parts d'un programa.

Consta d'un nom i paràmetres. Amb el nom cridem a la funció, és a dir, fem referència a la mateixa. Els paràmetres són valors que s'envien i són indispensables per a la resolució del mateix. La funció ha de fer alguna operació amb els paràmetres que li enviem. Podem carregar una variable, consultar-la, modificar-la, imprimir-la, etc.  
  
Fins i tot els programes més senzills tenen la necessitat de fragmentar-se. Les funcions són els únics tipus de subprogrames que accepta JavaScript. Tenen la següent estructura:

function <nom de funció> (argument 1, argument 2, ..., argument n){  
  <Codi de la funció>  
}

Hem de buscar un nom de funció que ens indiqui quin és el seu objectiu (Si la funció rep una cadena i el centra, potser hauríem de cridar-la centrarTitol). Veurem que una funció pot variar bastant en la seva estructura, pot tenir o no paràmetres, retornar un valor, etc.

Exemple: Mostra un missatge que es repeteixi 3 vegades a la pàgina amb el següent text:  
'Error: Escriu el teu document correctament'  
  
La solució sense utilitzar funcions és:

<!DOCTYPE html>

<html>

<head>

<title>Pràctica 11</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<script>

function inici(){

document.write('<h1>Pràctica 11</h1>');

document.write('Error: Escriu el teu document correctament<br>');

document.write('Error: Escriu el teu document correctament<br>');

document.write('Error: Escriu el teu document correctament<br>');

}

window.onload=inici;

</script>

</head>

<body>

</body>

</html>

Utilitzant una funció:

<!DOCTYPE html>

<html>

<head>

<title>Pràctica 11</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<script>

function inici(){

document.write('<h1>Pràctica 11</h1>');

mostraMissatge();

mostraMissatge();

mostraMissatge();

}

function mostraMissatge(){

document.write('Error: Escriu el teu document correctament<br>');

}

window.onload=inici;

</script>

</head>

<body>

</body>

</html>  
  
Recordem que JavaScript és sensible a majúscules i minúscules. Si fixem com a nom a la funció mostraMissatge (és a dir la segona paraula amb majúscula) hem de respectar aquest nom quan la cridem a aquesta funció.

És important notar que perquè una funció s'executi hem de cridar-la des de fora pel seu nom (en aquest exemple: mostraMissatge()).

Cada cop que es crida una funció s'executen totes les línies contingudes en la mateixa.  
Si no es crida la funció, les instruccions de la mateixa mai s'executaran.  
A una funció la podem cridar tantes vegades com necessitem.  
Les funcions ens estalvien escriure codi que es repeteix amb freqüència i permet que el nostre programa sigui més comprensible.

**Funciones amb paràmetres**

Explicarem amb un exemple, una funció que té dades d'entrada.  
Exemple: Confeccionar una funció que rebi dos nombres i mostri a la pàgina els valors compresos entre ells d'un en un. Carregar per teclat aquests dos valors.

<!DOCTYPE html>

<html>

<head>

<title>Pràctica 11</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<script>

function inici(){

var valor1 = parseInt(prompt( 'Introduïu valor inferior:'));

var valor2 = parseInt(prompt( 'Introduïu valor superior:'));

document.write('<h1>Funcions amb paràmetres</h1>');

mostraCompresos(valor1,valor2);

function mostraCompresos(x1,x2){

for (var i = x1; i <= x2; i ++){

document.write(i + ' ');

}

};

}

window.onload=inici;

</script>

</head>

<body>

</body>

</html>

El programa de JavaScript comença a executar-se on definim les variables ‘valor1’ i ‘valor2’ i no on es defineix la funció. Després de carregar els dos valors per teclat es crida a la funció ‘mostraCompresos()’ i li enviem les variables ‘valor1’ i ‘valor2’. Els paràmetres ‘x1’ i ‘x2’ reben els continguts de les variables ‘valor1’ i ‘valor 2’.  
És important notar que a la funció la podem cridar la quantitat de vegades que volguem.  
  
**PROBLEMES**

1. Elaborar una funció a la qual li enviem tres enters i mostri el menor.
2. Confeccionar una funció a la qual li enviï tres enters i els mostri ordenats de menys a més.

**Funcions que retornen un valor**

Són comuns els casos on una funció, després de fer un procés, retorni un valor.

Exemple 1: Confeccionar una funció que rebi un valor enter entre 1 i 5. Després retornar en català el valor rebut.

<!DOCTYPE html>

<html>

<head>

<title>Pràctica 11</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<script>

function inici(){

var valor = parseInt(prompt( 'Introdueix un número entre 1-5:'));

document.write('<h1>Funcions que retornen un valor</h1>');

var res = numeroCatala(valor);

document.write('El número introduït és el: ' + res);

function numeroCatala(x){

switch(x){

case 1: return 'un';

case 2: return 'dos';

case 3: return 'tres';

case 4: return 'quatre';

case 5: return 'cinc';

default: return 'número incorrecte';

}

};

}

window.onload=inici;

</script>

</head>

<body>

</body>

</html>  
  
Podem veure que el valor retornat per una funció ho indiquem per mitjà de la paraula clau ‘return’. Quan es crida la funció, hem d'assignar el nom de la funció a una variable, ja que la mateixa retorna un valor.  
Una funció pot tenir diversos paràmetres, però només pot retornar un únic valor.  
  
  
  
Exemple 2: Confeccionar una funció que rebi una data amb el format de dia, mes i any i retorni una cadena amb un format similar a: "Avui es 30 de setembre del 2018".

<!DOCTYPE html>

<html>

<head>

<title>Pràctica 11</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<script>

function inici(){

document.write('<h1>Funcions que retornen un valor</h1>');

function formateaData(dia, mes, any){

var s = 'Avui és ' + dia + ' de ';

switch (mes) {

case 1: s += 'gener';

break;

case 2: s += 'febrer';

break;

case 3: s += 'març';

break;

case 4: s += 'abril';

break;

case 5: s += 'maig';

break;

case 6: s += 'juny';

break;

case 7: s += 'juliol';

break;

case 8: s += 'agost';

break;

case 9: s += 'setembre';

break;

case 10: s += 'octubre';

break;

case 11: s += 'novembre';

break;

case 12: s += 'desembre';

break;

}

s += ' de ' + any;

return s;

}

document.write (formateaData (30,9,2018));

}

window.onload=inici;

</script>

</head>

<body>

</body>

</html>

</html>

Com podem observar, la funció retorna el literal que després s’imprimirà en la pàgina. Aquesta funció va concatenant els literals a la variable ‘s’ per obtenir el resultat desitjat. Aquesta variable ‘s’ és la que retorna la funció.

**PROBLEMES**

1. Codifica una funció que rebi tres números sencers i retorni el major d’ells.
2. Codifica una funció que rebi el valor del costat d’un quadrat i retorni el seu perímetre.
3. Codifica una funció que retorni la quantitat de dígits que té un número sencer positiu.
4. Codifica una funció que rebi tres números sencers i retorni la mitjana.
5. Codifica una funció que demani la càrrega de 5 valors per teclat i retorni la seva suma.