



Tecnólogo en Análisis y Desarrollo de
Software
Ficha

Funciones JS

SENA

MANUAL FUNCIONES JAVASCRIPT

BRAYAN ESTIVEN CARVAJAL PADILLA

Análisis y Desarrollo de Software

Ficha: 2899747



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

1.

Nombre de la función: saludo(psaludar)		Versión: 1.0
Descripción: Función que saluda		
saludar	Tipo de variable: Numérico	
Código:		
<pre>function saludo(psaludar){ let saludar=psaludar return saludar + " Parametro" }</pre>		<div>Hola Mundo JS Parametro</div> <div>Hola Mundo JS Expresion</div>

Nombre de la función: saludoExp(psaludar)		Versión: 2.0
Descripción: Función que saluda		
saludar	Tipo de variable: Numérico	
Código:		
<pre>const saludoExp=function(psaludar){ ... let saludar=psaludar ... return saludar + " Expresion" }</pre>		<div>Hola Mundo JS Parametro</div> <div>Hola Mundo JS Expresion</div>



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

2.

Nombre de la función: suma(pnumeroUno, pnumeroDos)		Versión: 1.0
Descripción: Función que suma dos números		
numeroUno, numeroDos,sumar		Tipo de variable: Numérico
Código:		
<pre>function suma(pnumeroUno, pnumeroDos) { let numeroUno = pnumeroUno; let numeroDos = pnumeroDos; let sumar = numeroUno + numeroDos; return sumar + " Parametro"; }</pre>		<div>5 Parametro</div> <div>5 Expresion</div> <div>> </div>

5 Parametro

5 Expresion

> |

Nombre de la función: sumaExp(pnumeroUno, pnumeroDos)		Versión: 2.0
Descripción: Función que suma dos números		
numeroUno, numeroDos, sumar		Tipo de variable: Numérico
Código:		
<pre>const sumaExp=function(pnumeroUno, pnumeroDos){ let numeroUno = pnumeroUno; let numeroDos = pnumeroDos; let sumar = numeroUno + numeroDos; return sumar + "Expresion"; }</pre>		

5 Parametro

5 Expresion

> |



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

3.

Nombre de la función: suma(pnumeroUno, pnumeroDos)		Versión: 1.0
Descripción: Función que suma dos numeros		
numeroUno, numeroDos, sumar		Tipo de variable: Numérico
Código:		
<pre>function suma(pnumeroUno, pnumeroDos) { ... let numeroUno = pnumeroUno ... let numeroDos = pnumeroDos ... let sumar = numeroUno + numeroDos ... return sumar + " Parametro"; }</pre>		<div>6 Parametro</div> <div>0 Parametro</div> <div>9 Parametro</div> <div>1 Parametro</div>

6 Parametro

0 Parametro

9 Parametro

1 Parametro

Nombre de la función: resta(pnumeroUno, pnumeroDos)		Versión: 1.0
Descripción: Función que muestra la diferencia de dos numeros		
numeroUno, numeroDos, restar	Tipo de variable: Numérico	
Código:		
<pre>function resta(pnumeroUno, pnumeroDos) { ... let numeroUno = pnumeroUno ... let numeroDos = pnumeroDos ... let restar = numeroUno - numeroDos ... return restar + " Parametro"; }</pre>		<div>6 Parametro</div> <div>0 Parametro</div> <div>9 Parametro</div> <div>1 Parametro</div>

6 Parametro

0 Parametro

9 Parametro

1 Parametro



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: multiplicacion(pnumeroUno, pnumeroDos)		Versión: 1.0
Descripción: Función que multiplica dos numeros		
numeroUno, numeroDos, multiplicar	Tipo de variable: Numérico	
Código:		
<pre>function multiplicacion(pnumeroUno, pnumeroDos) { ... let numeroUno = pnumeroUno ... let numeroDos = pnumeroDos ... let multiplicar = numeroUno * numeroDos ... return multiplicar + " Parametro"; }</pre>		<div>6 Parametro</div> <div>0 Parametro</div> <div>9 Parametro</div> <div>1 Parametro</div>

6 Parametro

0 Parametro

9 Parametro

1 Parametro

Nombre de la función: division(pnumeroUno, pnumeroDos)		Versión: 1.0
Descripción: Función que divide dos numeros		
numeroUno, numeroDos, dividir	Tipo de variable: Numérico	
Código:		
<pre>function division(pnumeroUno, pnumeroDos) { ... let numeroUno = pnumeroUno ... let numeroDos = pnumeroDos ... let dividir = numeroUno / numeroDos ... return dividir + " Parametro"; }</pre>		<div>6 Parametro</div> <div>0 Parametro</div> <div>9 Parametro</div> <div>1 Parametro</div>

6 Parametro

0 Parametro

9 Parametro

1 Parametro



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: sumaExp(pnumeroUno, pnumeroDos)		Versión: 2.0
Descripción: Funciones que suma dos numeros		
numeroUno, numeroDos, sumar	Tipo de variable: Numérico	
Código:		
<pre>const sumaExp= function(pnumeroUno, pnumeroDos) { ...let numeroUno = pnumeroUno ...let numeroDos = pnumeroDos ...let sumar = numeroUno + numeroDos ...return sumar + "Expresion"; }</pre>		<div>1 Parametro</div> <div>6 Expresion</div> <div>0 Expresion</div> <div>9 Expresion</div> <div>1 Expresion</div>

1 Parametro

6 Expresion

0 Expresion

9 Expresion

1 Expresion

Nombre de la función: restaExp(pnumeroUno, pnumeroDos)		Versión: 2.0
Descripción: Funciones muestra la diferencia de dos numeros		
numeroUno, numeroDos, restar	Tipo de variable: Numérico	
Código: <pre>const restaExp= function(pnumeroUno, pnumeroDos) { ...let numeroUno = pnumeroUno ...let numeroDos = pnumeroDos ...let restar = numeroUno - numeroDos ...return restar + " Expresion"; }</pre>		

1 Parametro

6 Expresion

0 Expresion

9 Expresion

1 Expresion

1 Parametro

6 Expresion

0 Expresion

9 Expresion

1 Expresion



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: multiplicacionExp(pnumeroUno, pnumeroDos)		Versión: 2.0
Descripción: Funciones que multiplica dos numeros		
numeroUno, numeroDos, multiplicar	Tipo de variable: Numérico	
Código:		
<pre>const multiplicacionExp= function(pnumeroUno, pnumeroDos) { ...let numeroUno = pnumeroUno ...let numeroDos = pnumeroDos ...let multiplicar = numeroUno * numeroDos ...return multiplicar + " Expresion"; }</pre>		<div>1 Parametro</div> <div>6 Expresion</div> <div>0 Expresion</div> <div>9 Expresion</div> <div>1 Expresion</div>

1 Parametro
6 Expresion
0 Expresion
9 Expresion
1 Expresion

Nombre de la función: divisionExp(pnumeroUno, pnumeroDos)		Versión: 2.0
Descripción: Funciones que divide dos numeros		
numeroUno, numeroDos, dividir		Tipo de variable: Numérico
Código: <pre>const divisionExp= function(pnumeroUno, pnumeroDos) { ...let numeroUno = pnumeroUno ...let numeroDos = pnumeroDos ...let dividir = numeroUno / numeroDos ...return dividir + "Expresion"; }</pre>		

1 Parametro

6 Expression

0 Expression

9 Expression

1 Expression

1 Parametro
6 Expresion
0 Expresion
9 Expresion
1 Expresion







Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

4.

Nombre de la función: porcentaje(pnumeroUno)		Versión: 1.0
Descripción: Función que da el porcentaje de un numero		
numeroUno, porcentaje	Tipo de variable: Numérico	
Código:		
<pre>function porcentaje(pnumeroUno) { let numeroUno = pnumeroUno let porcentaje = numeroUno / 100 return porcentaje + " Parametro"; }</pre>		    <u>0.03 Parametro</u>

Nombre de la función: porcentajeExp(pnumeroUno)		Versión: 2.0
Descripción: Función que da el porcentaje de un numero		
numeroUno, porcentaje	Tipo de variable: Numérico	
Código:		
<pre>const porcentajeExp= function(pnumeroUno) { let numeroUno = pnumeroUno let porcentaje = numeroUno / 100 return porcentaje + " Expresion"; }</pre>		0.03 Expresion



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

5.

Nombre de la función: prom(pnumrtoaUno, pnumeroDos, pnumeroTres)		Versión: 1.0
Descripción: Función que da el promedio de 3 notas		
numeroUno, numeroDos, numeroTres, prom	Tipo de variable: Numérico	
Código:		
<pre>function prom(pnumeroUno, pnumeroDos, pnumeroTres) { ... let numeroUno = pnumeroUno ... let numeroDos = pnumeroDos ... let numeroTres = pnumeroTres ... let prom = (numeroUno + numeroDos + numeroTres)/3 ... return prom + " Parametro"; }</pre>		

3.333333333333335 Parametro

Nombre de la función: promExp(pnumrtoaUno, pnumeroDos, pnumeroTres)		Versión: 2.0
Descripción: Función que da el promedio de 3 notas		
numeroUno, numeroDos, numeroTres, prom	Tipo de variable: Numérico	
Código:		
<pre>const promExp= function(pnumeroUno, pnumeroDos, pnumeroTres) { ... let numeroUno = pnumeroUno ... let numeroDos = pnumeroDos ... let numeroTres = pnumeroTres ... let prom = (numeroUno + numeroDos + numeroTres)/3 ... return prom + " Expresion"; }</pre>		
3.3333333333333335 Expresion		

3.333333333333335 Expresion



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

6.

Nombre de la función: calcPorcen(pnota, pporcent)		Versión: 1.0
Descripción: Función que da el porcentaje de 3 notas		
nota, porcent, resultado	Tipo de variable: Numérico	
Código:		
<pre>function calcPorcen(pnota, pporcent) { ... let nota=pnota ... let porcent=pporcent ... resultado = (nota * porcent) / 100 ... return resultado }</pre>		
Con Parametros:		
Porcentaje nota 1: 0.99		
Porcentaje nota 2: 1.5		
Porcentaje nota 3: 1.12		
La sumatoria de los porcentajes: 3.6100000000000003		

Nombre de la función: calcPorcenExp(pnota, pporcent)		Versión: 2.0
Descripción: Función que da el porcentaje de 3 notas		
nota, porcent, resultado	Tipo de variable: Numérico	
Código:		
<pre>const calcPorcenExp= function(pnota, pporcent) { ... let nota=pnota ... let porcent=pporcent ... resultado = (nota * porcent) / 100 ... return resultado }</pre>		
Con Expresion:		
Porcentaje nota 1: 0.99		
Porcentaje nota 2: 1.5		
Porcentaje nota 3: 1.12		
La sumatoria de los porcentajes: 3.6100000000000003		



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

7.

Nombre de la función: areaFiguras(pbase, paltura)		Versión: 1.0
Descripción: Función que da el área del triángulo, rectángulo y cuadrado		
base, altura, resultado	Tipo de variable: Numérico	
Código:		
<pre>function areaFiguras(pbase, paltura) { ... let base=pbase ... let altura=paltura ... let resultado = (base * altura) ... return resultado }</pre>		<div>Con Parametros:</div> <div>Area Cuadrado: 9</div> <div>Area Triangulo: 15</div> <div>Area Rectangulo: 28</div>

Nombre de la función: areaFigurasExp(pbase, paltura)		Versión: 2.0
Descripción: Función que da el área del triángulo, rectángulo y cuadrado		
base, altura, resultado		Tipo de variable: Numérico
Código:		
<pre>const areaFigurasExp= function(pbase, paltura) { ... let base=pbase ... let altura=paltura ... let resultado = (base * altura) ... return resultado }</pre>		
<div>Con Experesion: Area Cuadrado: 9 Area Triangulo: 15 Area Rectangulo: 28</div>		



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

8.

Nombre de la función: salario(pdias, pVdia)		Versión: 1.0
Descripción: Función que calcula el salario de una persona		
dias, Vdia, totalSalario		Tipo de variable: Numérico
Código:		
<pre>function salario(pdias, pVdia) { let dias = pdias let Vdia = pVdia let totalSalario totalSalario = dias * Vdia return totalSalario }</pre>		<div>PARAMETRO</div> <div>Salario 250000</div> <div>Subsidio Transporte 140000</div> <div>Pago Total 307000</div>

PARAMETRO

Salario	250000
Subsidio Transporte	140000
Pago Total	307000

Nombre de la función: SubTrans(pdias, pVdia)		Versión: 1.0				
Descripción: Función que calcula el subsidio de transporte de una persona						
SalarioMinimo, SalarioTrans, SubTransporte	Tipo de variable: Numérico					
Código:						
<pre>function SubTrans(pdias, pVdia) { let SalarioMinimo = 1600000 let SalarioTrans = salario(pdias, pVdia) let SubTransporte if(SalarioTrans<=2*SalarioMinimo){ SubTransporte=140000 } else{ SubTransporte=0 } return SubTransporte }</pre>		<table><tr><th>PARAMETRO</th></tr><tr><td>Salario 250000</td></tr><tr><td>Subsidio Transporte 140000</td></tr><tr><td>Pago Total 307000</td></tr></table>	PARAMETRO	Salario 250000	Subsidio Transporte 140000	Pago Total 307000
PARAMETRO						
Salario 250000						
Subsidio Transporte 140000						
Pago Total 307000						

PARAMETRO

Salario	250000
Subsidio Transporte	140000
Pago Total	307000



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: salud(pdias, pVdia)		Versión: 1.0				
Descripción: Función que calcula el costo salud de una persona						
pagoSalud	Tipo de variable: Numérico					
Código:						
<pre>function salud(pdias, pVdia) { let pagoSalud pagoSalud = salario(pdias, pVdia) * 0.12 return pagoSalud }</pre>		<table><tr><th>PARAMETRO</th></tr><tr><td>Salario 250000</td></tr><tr><td>Subsidio Transporte 140000</td></tr><tr><td>Pago Total 307000</td></tr></table>	PARAMETRO	Salario 250000	Subsidio Transporte 140000	Pago Total 307000
PARAMETRO						
Salario 250000						
Subsidio Transporte 140000						
Pago Total 307000						

Nombre de la función: pension(pdias, pVdia)		Versión: 1.0				
Descripción: Función que calcula el costo pensión de una persona						
pagoPension	Tipo de variable: Numérico					
Código:						
<pre>function pension(pdias, pVdia) { let pagoPension pagoPension = salario(pdias, pVdia) * 0.16 return pagoPension }</pre>		<table><tr><th>PARAMETRO</th></tr><tr><td>Salario 250000</td></tr><tr><td>Subsidio Transporte 140000</td></tr><tr><td>Pago Total 307000</td></tr></table>	PARAMETRO	Salario 250000	Subsidio Transporte 140000	Pago Total 307000
PARAMETRO						
Salario 250000						
Subsidio Transporte 140000						
Pago Total 307000						

Nombre de la función: arl(pdias, pVdia)		Versión: 1.0				
Descripción: Función que calcula el costo arl de una persona						
pagoArl	Tipo de variable: Numérico					
Código:						
<pre>function arl(pdias, pVdia) { let pagoArl pagoArl = salario(pdias, pVdia) * 0.052 return pagoArl }</pre>		<table><tr><th>PARAMETRO</th></tr><tr><td>Salario 250000</td></tr><tr><td>Subsidio Transporte 140000</td></tr><tr><td>Pago Total 307000</td></tr></table>	PARAMETRO	Salario 250000	Subsidio Transporte 140000	Pago Total 307000
PARAMETRO						
Salario 250000						
Subsidio Transporte 140000						
Pago Total 307000						



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: deducible(pdias, pVdia)		Versión: 1.0
Descripción: Función que calcula el descuento del pago total de una persona		
pagoDeducible		Tipo de variable: Numérico
Código: <pre>function deducible(pdias, pVdia) { let pagoDeducible pagoDeducible = pension(pdias, pVdia) + salud(pdias, pVdia) + arl(pdias, pVdia) return pagoDeducible }</pre>		
PARAMETRO		
Salario 250000		
Subsidio Transporte 140000		
Pago Total 307000		

Nombre de la función: pagoTotal(pdias, pVdia)		Versión: 1.0
Descripción: Función que calcula el pago total de una persona		
pagoSueldo		Tipo de variable: Numérico
Código:		
<pre>function pagoTotal(pdias, pVdia) { let pagoSueldo pagoSueldo = salario(pdias, pVdia) + SubTrans(pdias, pVdia) - deducible(pdias, pVdia) return pagoSueldo }</pre>		
PARAMETRO		
Salario 250000		
Subsidio Transporte 140000		
Pago Total 307000		



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: salarioExp(pdias, pVdia)		Versión: 2.0
Descripción:		
Función que calcula el salario de una persona		
días, Vdia, totalSalario		Tipo de variable: Numérico
Código:		
<pre>const salarioExp = function (pdias, pVdia) { let dias = pdias let Vdia = pVdia let totalSalario totalSalario = dias * Vdia return totalSalario }</pre>		<div>EXPRESION</div> <div>Salario 250000</div> <div>Subsidio Transporte 140000</div> <div>Pago Total 307000</div>

EXPRESION

Salario 250000

Subsidio Transporte 140000

Pago Total 307000

Nombre de la función: SubTransExp(pdias, pVdia)		Versión: 2.0
Descripción: Función que calcula el subsidio de transporte de una persona		
SalarioMinimo, SalarioTrans, SubTransporte		Tipo de variable: Numérico
Código:		
<pre>const SubTransExp = function (pdias, pVdia) { let SalarioMinimo = 1600000 let SalarioTrans = salario(pdias, pVdia) let SubTransporte if(SalarioTrans<=2*SalarioMinimo){ SubTransporte=140000 } else{ SubTransporte=0 } return SubTransporte }</pre>		<div>EXPRESION</div> <div>Salario 250000</div> <div>Subsidio Transporte 140000</div> <div>Pago Total 307000</div>

EXPRESION

Salario 250000

Subsidio Transporte 140000

Pago Total 307000



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: saludExp(pdias, pVdia)		Versión: 2.0
Descripción: Función que calcula el costo salud de una persona		
pagoSalud	Tipo de variable: Numérico	
Código:		
<pre>const saludExp = function (pdias, pVdia) { let pagoSalud pagoSalud = salario(pdias, pVdia) * 0.12 return pagoSalud }</pre>		<div>EXPRESION</div> <div>Salario 250000</div> <div>Subsidio Transporte 140000</div> <div>Pago Total 307000</div>

Nombre de la función: pensionExp(pdias, pVdia)		Versión: 2.0
Descripción: Función que calcula el costó pensión de una persona		
pagoPension,	Tipo de variable: Numérico	
Código:		
<pre>const pensionExp = function (pdias, pVdia) { let pagoPension pagoPension = salario(pdias, pVdia) * 0.16 return pagoPension }</pre>		<div>EXPRESION</div> <div>Salario 250000</div> <div>Subsidio Transporte 140000</div> <div>Pago Total 307000</div>

Nombre de la función: arlExp(pdias, pVdia)		Versión: 2.0
Descripción: Función que calcula el costó arl de una persona		
pagoArl	Tipo de variable: Numérico	
Código: <pre>const arlExp = function (pdias, pVdia) { let pagoArl pagoArl = salario(pdias, pVdia) * 0.052 return pagoArl }</pre>		
		EXPRESION
		Salario 250000
		Subsidio Transporte 140000
		Pago Total 307000



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: deducibleExp(pdias, pVdia)		Versión: 2.0
Descripción: Función que calcula el descuento del pago total de una persona		
pagoDeducible		Tipo de variable: Numérico
Código: <pre>const deducibleExp = function (pdias, pVdia) { let pagoDeducible pagoDeducible = pension(pdias, pVdia) + salud(pdias, pVdia) + arl(pdias, pVdia) return pagoDeducible }</pre>		
EXPRESION		
Salario 250000		
Subsidio Transporte 140000		
Pago Total 307000		

Nombre de la función: pagoTotalExp(pdias, pVdia)		Versión: 2.0
Descripción: Función que calcula el pago total de una persona		
pagoSueldo		Tipo de variable: Numérico
Código:		
<pre>const pagoTotalExp = function (pdias, pVdia) { let pagoSueldo pagoSueldo = salario(pdias, pVdia) + SubTrans(pdias, pVdia) - deducible(pdias, pVdia) return pagoSueldo }</pre>		
EXPRESION		
Salario 250000		
Subsidio Transporte 140000		
Pago Total 307000		



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

9.

Nombre de la función: edadPersona(pedad)		Versión: 1.0
Descripción: Función que identifica si una persona es mayor o menor de edad		
edad	Tipo de variable: Numérico	
Código:		

```
function edadPersona(pedad) {  
  let edad=pedad  
  if(edad>=18){  
    console.log("Es mayor de edad")  
  }else{  
    console.log("Es menor de edad")  
  }  
  return edad  
}
```

Con Parametros:

Es mayor de edad

19

Con Parametros:
Es mayor de edad
19

Nombre de la función: edadPersonaExp(pedad)		Versión: 2.0
Descripción: Función que identifica si una persona es mayor o menor de edad		
edad	Tipo de variable: Numérico	
Código:		
<pre>const edadPersonaExp= function(pedad) { let edad=pedad if(edad>=18){ console.log("Es mayor de edad") }else{ console.log("Es menor de edad") } return edad }</pre>		<div>Con Expresion:</div> <div>Es menor de edad</div> <div>17</div>

Con Expresion:
Es menor de edad
17



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

10.

Nombre de la función: calcEdad(panoAct, panoNac)		Versión: 1.0
Descripción: Función que calcula la edad de una persona		
anoAct, anoNac, edad		Tipo de variable: Numérico
Código:		
<pre>function calcEdad(panoAct, panoNac) { ... let anoAct = panoAct ... let anoNac = panoNac ... let edad = anoAct - anoNac ... if(edad>17){ ... console.log("Es mayor de edad"); ... }else{ ... console.log("Es menor de edad"); ... } ... return edad }</pre>		<div>Con Parametros:</div> <div>Es mayor de edad</div> <div>19</div>

Con Parametros:

Es mayor de edad

19

Nombre de la función: calcEdadExp(panoAct, panoNac)		Versión: 2.0
Descripción: Función que calcula la edad de una persona		
anoAct, anoNac, edad	Tipo de variable: Alfanumérico	
Código:		

```
const calcEdadExp = function (panoAct, panoNac) {  
  ... let anoAct = panoAct  
  ... let anoNac = panoNac  
  ... let edad = anoAct - anoNac  
  ... if(edad>17){  
    ... console.log("Es mayor de edad");  
  ... }else{  
    ... console.log("Es menor de edad");  
  ... }  
  ... return edad  
}
```

Con Expresion:

Es menor de edad

16

Con Expresion:

Es menor de edad

16



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

11.

Nombre de la función: numCalc(pnum1, pnum2)		Versión: 1.0
Descripción: Función que calcula que numero es mayor		
num1, num2	Tipo de variable: Numérico	
Código:		
<pre>function numCalc(pnum1, pnum2){ let num1 = pnum1 let num2 = pnum2 if(num1==num2){ console.log("Son iguales"); } else{ if(num1>num2){ console.log("El numero uno es mayor"); }else{ console.log("El numero dos es mayor"); } } return "Num1: "+num1+" Num2: "+num2 }</pre>		<div>Con Parametros:</div> <div>El numero dos es mayor</div> <div>Num1: 3 Num2: 8</div>

Con Parametros:

El numero dos es mayor

Num1: 3 Num2: 8



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: numCalcExp(pnum1, pnum2)

Versión: 2.0

Descripción:

Función que calcula que numero es mayor

num1, num2

Tipo de variable: Numérico

Código:

```
const numCalcExp = function (pnum1, pnum2) {  
  let num1 = pnum1  
  let num2 = pnum2  
  if (num1 == num2) {  
    console.log("Son iguales");  
  }  
  else {  
    if (num1 > num2) {  
      console.log("El numero uno es mayor");  
    } else {  
      console.log("El numero dos es mayor");  
    }  
  }  
  return "Num1: " + num1 + " Num2: " + num2  
}
```

Con Expresion:

El numero uno es mayor

Num1: 45 Num2: 13



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

12.

Nombre de la función: areaMay(plad1, plad2, plad3)		Versión: 1.0
Descripción: Función que calcula el área mayor entre 3 cuadrados		
lad1, lad2, lad3, cua1, cua2, cua3	Tipo de variable: Numérico	
Código:		
<pre>function areaMay(plad1, plad2, plad3) { ... let lad1 = plad1 ... let lad2 = plad2 ... let lad3 = plad3 ... cua1 = lad1 * lad1 ... cua2 = lad2 * lad2 ... cua3 = lad3 * lad3 ... if (cua1 == cua2 && cua2 == cua3) { ... console.log("Todos las areas son iguales"); ... } ... else { ... if (cua1 > cua2 && cua1 > cua3) { ... console.log("El area uno es mayor") ... } ... else { ... if (cua2 > cua1 && cua2 > cua3) { ... console.log("El area dos es mayor") ... } else { ... console.log("El area tres es mayor") ... } ... } ... } ... return "Area1: " + cua1 + " Area2: " + cua2 + " Area3: " + cua3 }</pre>		<div>Con Parametros:</div> <div>El area dos es mayor</div> <div>Area1: 9 Area2: 64 Area3: 16</div>



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: areaMayExp(plad1, plad2, plad3)

Versión: 2.0

Descripción:

Función que calcula el área mayor entre 3 cuadrados

lad1, lad2, lad3,
cua1, cua2, cua3

Tipo de variable: Numérico

Código:

```
const areaMayExp = function (plad1, plad2, plad3) {  
  ... let lad1 = plad1  
  ... let lad2 = plad2  
  ... let lad3 = plad3  
  ... cua1 = lad1 * lad1  
  ... cua2 = lad2 * lad2  
  ... cua3 = lad3 * lad3  
  ... if (cua1 == cua2 && cua2 == cua3) {  
  ...   ... console.log("Todos las areas son iguales");  
  ... }  
  ... else {  
  ...   ... if (cua1 > cua2 && cua1 > cua3) {  
  ...     ... console.log("El area uno es mayor")  
  ...   }  
  ...   ... else {  
  ...     ... if (cua2 > cua1 && cua2 > cua3) {  
  ...       ... console.log("El area dos es mayor")  
  ...     } else {  
  ...       ... console.log("El area tres es mayor")  
  ...     }  
  ...   }  
  ... }  
  ... return "Area1: " + cua1 + " Area2: " + cua2 + " Area3: " + cua3  
}
```

Con Expresion:

El area uno es mayor

Area1: 2304 Area2: 225 Area3: 961



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

13.

Nombre de la función: edadPerson(panoAct1, panoNac1, panoAct2, panoNac2, panoAct3, panoNac3)	Versión: 1.0
Descripción: Función que imprime la edad, el promedio de edades y el promedio mayor de edad entre 3 personas	
panoAct1, panoNac1, panoAct2, panoNac2, panoAct3, panoNac3, edad1, edad2, edad3, prom	Tipo de variable: Numérico
Código: <pre>function edadPerson(panoAct1, panoNac1, panoAct2, panoNac2, panoAct3, panoNac3) { ...let anoAct1 = panoAct1, anoAct2 = panoAct2, anoAct3 = panoAct3 ...let anoNac1 = panoNac1, anoNac2 = panoNac2, anoNac3 = panoNac3 ...edad1 = anoAct1 - anoNac1 ...edad2 = anoAct2 - anoNac2 ...edad3 = anoAct3 - anoNac3 ...prom = (edad1 + edad2 + edad3) / 3 ...if (edad1 >= 18) { console.log("La primera edad es mayor") ...} else { console.log("La primera edad es menor") ...} ...if (edad2 >= 18) { console.log("La segunda edad es mayor") ...} else { console.log("La segunda edad es menor") ...} ...if (edad3 >= 18) { console.log("La tercera edad es mayor") ...} else { console.log("La tercera edad es menor") ...} ...console.log("El promedio de edad es: " + prom) ...if (prom < 18) { console.log("El promedio es minoria de edad") ...} else { console.log("El promedio es mayoria de edad") ...} ...return "edad1: " + edad1 + " edad2: " + edad2 + " edad3: " + edad3 }</pre>	

Con Parametros:

La primera edad es mayor

La segunda edad es mayor

La tercera edad menor

El promedio de edad es: 18

El promedio es mayoria de edad

edad1: 19 edad2: 18 edad3: 17



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: edadPersonExp(panoAct1, panoNac1, panoAct2, panoNac2, panoAct3, panoNac3)	Versión: 2.0
--	---------------------

Descripción:

Función que imprime la edad, el promedio de edades y el promedio mayor de edad entre 3 personas

panoAct1, panoNac1, panoAct2, panoNac2, panoAct3, panoNac3, edad1, edad2, edad3, prom	Tipo de variable: Numérico
---	----------------------------

Código:

```
const edadPersonExp = function (panoAct1, panoNac1, panoAct2, panoNac2, panoAct3, panoNac3) {  
  ...let anoAct1 = panoAct1, anoAct2 = panoAct2, anoAct3 = panoAct3  
  ...let anoNac1 = panoNac1, anoNac2 = panoNac2, anoNac3 = panoNac3  
  ...edad1 = anoAct1 - anoNac1  
  ...edad2 = anoAct2 - anoNac2  
  ...edad3 = anoAct3 - anoNac3  
  ...prom = (edad1 + edad2 + edad3) / 3  
  ...if (edad1 >= 18) {  
  ...  ...console.log("La primera edad es mayor")  
  ...} else {  
  ...  ...console.log("La primera edad es menor")  
  ...}  
  ...if (edad2 >= 18) {  
  ...  ...console.log("La segunda edad es mayor")  
  ...} else {  
  ...  ...console.log("La segunda edad es menor")  
  ...}  
  ...if (edad3 >= 18) {  
  ...  ...console.log("La tercera edad es mayor")  
  ...} else {  
  ...  ...console.log("La tercera edad es menor")  
  ...}  
  ...console.log("El promedio de edad es: " + prom)  
  ...if (prom < 18) {  
  ...  ...console.log("El promedio es minoria de edad")  
  ...} else {  
  ...  ...console.log("El promedio es mayoria de edad")  
  ...}  
  ...return "edad1: " + edad1 + " edad2: " + edad2 + " edad3: " + edad3  
}
```

Con Expresion:

La primera edad es mayor

La segunda edad es menor

La tercera edad es mayor

El promedio de edad es: 29

El promedio es mayoria de edad

edad1: 55 edad2: 11 edad3: 21



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

14.

Nombre de la función: proNotas(pnota1, pnota2, pnota3)		Versión: 1.0					
Descripción: Función que calcula el porcentaje de 3 notas y su sumatoria							
nota1, nota2, nota3, percent1, percent2, percent3, sum	Tipo de variable: Numérico						
Código:							
<pre>function proNotas(pnota1, pnota2, pnota3) { let nota1 = pnota1 let nota2 = pnota2 let nota3 = pnota3 percent1 = (nota1 * 20) / 100 percent2 = (nota2 * 35) / 100 percent3 = (nota3 * 45) / 100 sum = percent1 + percent2 + percent3 if (sum > 4.5) { console.log("Nota superior") } else { if (sum <= 4.5 && sum > 3.5) { console.log("Nota buena") } else { if (sum <= 3.5 && sum >= 3) { console.log("Nota media") } else { console.log("Nota mala") } } } console.log("El 20% de la nota1 es: " + percent1) console.log("El 35% de la nota2 es: " + percent2) console.log("El 45% de la nota3 es: " + percent3) return "Suma de los porcentajes: " + sum }</pre>		<div>Con Parametros:</div> <table><tr><td>Nota buena</td></tr><tr><td>El 20% de la nota1 es: 0.48</td></tr><tr><td>El 35% de la nota2 es: 1.33</td></tr><tr><td>El 45% de la nota3 es: 1.8</td></tr><tr><td>Suma de los porcentajes: 3.6100000000000003</td></tr></table>	Nota buena	El 20% de la nota1 es: 0.48	El 35% de la nota2 es: 1.33	El 45% de la nota3 es: 1.8	Suma de los porcentajes: 3.6100000000000003
Nota buena							
El 20% de la nota1 es: 0.48							
El 35% de la nota2 es: 1.33							
El 45% de la nota3 es: 1.8							
Suma de los porcentajes: 3.6100000000000003							

Con Parametros:

Nota buena

El 20% de la nota1 es: 0.48

El 35% de la nota2 es: 1.33

El 45% de la nota3 es: 1.8

Suma de los porcentajes: 3.6100000000000003



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: proNotasExp(pnota1, pnota2, pnota3)

Versión: 2.0

Descripción:

Función que calcula el porcentaje de 3 notas y su sumatoria

nota1, nota2, nota3, percent1, percent2, percent3, sum

Tipo de variable: Numérico

Código:

```
const proNotasExp = function (pnota1, pnota2, pnota3) {  
  ... let nota1 = pnota1  
  ... let nota2 = pnota2  
  ... let nota3 = pnota3  
  ... percent1 = (nota1 * 20) / 100  
  ... percent2 = (nota2 * 35) / 100  
  ... percent3 = (nota3 * 45) / 100  
  ... sum = percent1 + percent2 + percent3  
  ... if (sum > 4.5) {  
  ...   ... console.log("Nota superior")  
  ... }  
  ... else {  
  ...   ... if (sum <= 4.5 && sum > 3.5) {  
  ...     ... console.log("Nota buena")  
  ...   ... }  
  ...   ... else {  
  ...     ... if (sum <= 3.5 && sum >= 3) {  
  ...       ... console.log("Nota media")  
  ...     ... }  
  ...     ... else {  
  ...       ... console.log("Nota mala")  
  ...     ... }  
  ...   ... }  
  ...   ... console.log("El 20% de la nota1 es: " + percent1)  
  ...   ... console.log("El 35% de la nota2 es: " + percent2)  
  ...   ... console.log("El 45% de la nota3 es: " + percent3)  
  ... }  
  ... return "Suma de los porcentajes: " + sum  
}
```

Con Expresion:

Nota mala

El 20% de la nota1 es: 0.42

El 35% de la nota2 es: 0.665

El 45% de la nota3 es: 1.395

Suma de los porcentajes: 2.48



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

15-16.

Nombre de la función: contWhile(pcontar, pnumero)		Versión: 1.0
Descripción: Función que cuenta del 1 hasta el 5 con WHILE		
contar, numero	Tipo de variable: Numérico	
Código: <pre>//WHILE Con Parametro function contWhile(pcontar, pnumero) { ... let contar = pcontar ... let numero = pnumero ... while (contar <= numero) { contar = contar + 1; console.log(contar); ... } ... return "Se conto hasta " + contar }</pre>		

WHILE Con Parametros:

1

2

3

4

5

Se conto hasta 5

WHILE Con Parametros:

1

2

3

4

5

Se conto hasta 5

Nombre de la función: contFor(pcontar, pnumero)		Versión: 1.0
Descripción: Función que cuenta del 1 hasta el 5 con FOR		
contar, numero	Tipo de variable: Numérico	
Código:		
<pre>//FOR Con Parametro function contFor(pcontar, pnumero){ ...let contar = pcontar ...let numero = pnumero ...for (contar; contar <= numero; contar++){ console.log(contar); ...} ...return "Se conto hasta " + contar }</pre>		<pre>FOR Con Parametros: 1 2 3 4 5 Se conto hasta 6</pre>

FOR Con Parametros:

1

2

3

4

5

Se conto hasta 6



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: contWhileExp(pcontar, pnumero)

Versión: 2.0

Descripción:

Función que cuenta del 1 hasta el 5 con WHILE

contar, numero

Tipo de variable: Numérico

Código:

```
//WHILE Con Expresion
const contWhileExp = function (pcontar, pnumero) {
  ... let contar = pcontar
  ... let numero = pnumero
  ... while (contar < numero) {
  ...   ... contar = contar + 1;
  ...   ... console.log(contar);
  ... }
  ... return "Se conto hasta " + contar
}
```

WHILE Con Expresion:

1

2

3

4

5

Se conto hasta 5

Nombre de la función: contForExp(pcontar, pnumero)

Versión: 2.0

Descripción:

Función que cuenta del 1 hasta el 5 con FOR

contar, numero

Tipo de variable: Numérico

Código:

```
//FOR Con Expresion
const contForExp = function (pcontar, pnumero) {
  ... let contar = pcontar
  ... let numero = pnumero
  ... for (contar; contar <= numero; contar++) {
  ...   ... console.log(contar);
  ... }
  ... return "Se conto hasta " + contar
}
```

FOR Con Expresion:

1

2

3

4

5

Se conto hasta 6



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

17-18.

Nombre de la función: esParImpar(num)		Versión: 1.0
Descripción: Función que dice si el numero dado es par o impar		
num	Tipo de variable: Numérico	
Código:		
<pre>function esParImpar(num) { if (num % 2 === 0) { return "es par" } else { return "es impar" } }</pre>		

Nombre de la función: cicloWhile(pcontar, pnumero)		Versión: 1.0
Descripción: Función que dice si los números contados son par o impar		
Contar, numero, parImpar	Tipo de variable: Numérico	
Código:		

```
function cicloWhile(pcontar, pnumero) {  
  ...let contar = pcontar  
  ...let numero = pnumero  
  ...let parImpar  
  ...while (contar < numero) {  
    ...parImpar = esParImpar(contar)  
    ...contar = contar + 1  
    ...if (contar % 2 == 0) {  
    ...console.log(contar + parImpar)  
    ...}  
    ...else {  
    ...console.log(contar + parImpar)  
    ...}  
    ...}  
  ...return ""  
}
```

WHILE Con Parametros:

1 es par

2 es impar

3 es par

4 es impar

5 es par



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: cicloFor(pcontar, pnumero)

Versión: 1.0

Descripción:

Función que dice si los números contados son par o impar

Contar, numero, parImpar

Tipo de variable: Numérico

Código:

```
function cicloFor(pcontar, pnumero) {  
  let contar = pcontar  
  let numero = pnumero  
  let parImpar  
  for (contar; contar <= numero; contar++) {  
    parImpar = esParImpar(contar)  
    if (contar % 2 == 0) {  
      console.log(contar + parImpar)  
    }  
    else {  
      console.log(contar + parImpar)  
    }  
  }  
  return ""  
}
```

FOR Con Parametros:

1 es impar

2 es par

3 es impar

4 es par

5 es impar



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: cicloWhileExp(pcontar, pnumero)

Versión: 2.0

Descripción:

Función que identifica los números pares e impares

Contar, numero, parImpar

Tipo de variable: Numérico

Código:

```
const cicloWhileExp = function (pcontar, pnumero) {  
  ... let contar = pcontar  
  ... let numero = pnumero  
  ... let parImpar  
  ... while (contar < numero) {  
    ... parImpar = esParImpar(contar)  
    ... contar = contar + 1  
    ... if (contar % 2 == 0) {  
      ... console.log(contar + parImpar)  
    ... }  
    ... else {  
      ... console.log(contar + parImpar)  
    ... }  
    ... }  
  ... return ""  
}
```

WHILE Con Expresion:

1 es par
2 es impar
3 es par
4 es impar
5 es par



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: cicloForExp(pcontar, pnumero)

Versión: 2.0

Descripción:

Función que identifica los números pares e impares

Contar, numero, parImpar

Tipo de variable: Numérico

Código:

```
const cicloForExp = function(pcontar, pnumero) {  
  ...let contar = pcontar  
  ...let numero = pnumero  
  ...let parImpar  
  ...for(contar; contar <= numero; contar++) {  
    ...parImpar = esParImpar(contar)  
    ...if(contar % 2 == 0) {  
      ...console.log(contar + parImpar)  
    ...}  
    ...else {  
      ...console.log(contar + parImpar)  
    ...}  
  ...}  
  ...return ""  
}
```

FOR Con Expresion:

1 es impar

2 es par

3 es impar

4 es par

5 es impar



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

19-20.

Nombre de la función: cicloWhile(pcont, pnum)		Versión: 1.0
Descripción: Función que muestra la table de multiplicar del 5 con WHILE		
Cont, num, mult	Tipo de variable: Numérico	
Código:		
<pre>//WHILE Con Parametro function cicloWhile(pcont, pnum){ ... let cont = pcont ... let num = pnum ... while(cont<=num){ mult=num*cont console.log(num+" * "+cont+" = "+mult); cont=cont+1 ... } ... return "Se conto hasta " + cont }</pre>		
<div>WHILE Con Expresion:</div> <div>5 * 1 = 5</div> <div>5 * 2 = 10</div> <div>5 * 3 = 15</div> <div>5 * 4 = 20</div> <div>5 * 5 = 25</div> <div>Se conto hasta 6</div>		

WHILE Con Expresion:

5 * 1 = 5

5 * 2 = 10

5 * 3 = 15

5 * 4 = 20

5 * 5 = 25

Se conto hasta 6

Nombre de la función: cicloFor(pcont, pnum)		Versión: 1.0
Descripción: Función que muestra la table de multiplicar del 5 con FOR		
Cont, num, mult	Tipo de variable: Numérico	
Código:		
<pre>//FOR Con Parametro function cicloFor(pcont, pnum){ let cont = pcont let num = pnum for (cont; cont<=num; cont++){ mult=num*cont console.log(num+" * "+cont+" = "+mult); } return "Se conto hasta " + cont }</pre>		
<div>FOR Con Parametros:</div> <div>5 * 1 = 5</div> <div>5 * 2 = 10</div> <div>5 * 3 = 15</div> <div>5 * 4 = 20</div> <div>5 * 5 = 25</div> <div>Se conto hasta 6</div>		

FOR Con Parametros:

5 * 1 = 5

5 * 2 = 10

5 * 3 = 15

5 * 4 = 20

5 * 5 = 25

Se conto hasta 6



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: cicloWhileExp(pcont, pnum)

Versión: 2.0

Descripción:

Función que muestra la table de multiplicar del 5 con While

Cont, num,
mult

Tipo de variable: Numérico

Código:

```
//WHILE Con Expresion
const cicloWhileExp = function (pcont, pnum) {
  ... let cont = pcont
  ... let num = pnum
  ... while(cont<=num){
  ...   ... mult=num*cont
  ...   ... console.log(num+" * "+cont+" = "+mult);
  ...   ... cont=cont+1
  ... }
  ... return "Se conto hasta " + cont
}
```

WHILE Con Expresion:

5 * 1 = 5

5 * 2 = 10

5 * 3 = 15

5 * 4 = 20

5 * 5 = 25

Se conto hasta 6

Nombre de la función: cicloForExp(pcont, pnum)

Versión: 2.0

Descripción:

Función que muestra la table de multiplicar del 5 con For

Cont, num,
mult

Tipo de variable: Numérico

Código:

```
//FOR Con Expresion
const cicloForExp = function (pcont, pnum) {
  ... let cont = pcont
  ... let num = pnum
  ... for (cont; cont<=num; cont++){
  ...   ... mult=num*cont
  ...   ... console.log(num+" * "+cont+" = "+mult);
  ... }
  ... return "Se conto hasta " + cont
}
```

FOR Con Expresion:

5 * 1 = 5

5 * 2 = 10

5 * 3 = 15

5 * 4 = 20

5 * 5 = 25

Se conto hasta 6



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

21-22.

Nombre de la función: esParImpar(num)		Versión: 1.0
Descripción: Función que dice si el numero dado es par o impar		
num	Tipo de variable: Numérico	
Código:		
<pre>function esParImpar(num) { if (num % 2 === 0) { return "par" } else { return "impar" } }</pre>		

Nombre de la función: cicloWhile(pcont, pnum)		Versión: 1.0
Descripción: Función que muestra la tabla de multiplicar del 9 hasta el 5 y dice si el resultado es par o impar con WHILE		
Cont, num, mult, parImpar	Tipo de variable: Numérico	
Código:		

```
function cicloWhile(pcont, pnum) {  
  let cont = pcont  
  let num = pnum  
  while (cont <= 5) {  
    let mult = num * cont  
    let parImpar = esParImpar(mult)  
    console.log(num + " * " + cont + " = " + mult + " es " + parImpar)  
    cont = cont + 1  
  }  
  return ""  
}
```

WHILE Con Parametros:
9 * 1 = 9 es impar
9 * 2 = 18 es par
9 * 3 = 27 es impar
9 * 4 = 36 es par
9 * 5 = 45 es impar

WHILE Con Parametros:

9 * 1 = 9 es impar

9 * 2 = 18 es par

9 * 3 = 27 es impar

9 * 4 = 36 es par

9 * 5 = 45 es impar



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: cicloFor(pcont, pnum)		Versión: 1.0
Descripción: Función que muestra la tabla de multiplicar del 9 hasta el 5 y dice si el resultado es par o impar con FOR		
Cont, num, mult, parImpar	Tipo de variable: Numérico	
Código: <pre>function cicloFor(pcont, pnum) { let cont = pcont let num = pnum for (cont; cont <= 5; cont++) { let mult = num * cont let parImpar = esParImpar(mult) console.log(num+" * "+cont+" = "+mult+" es "+parImpar) } return "" }</pre>		

FOR Con Parametros:

9 * 1 = 9 es impar

9 * 2 = 18 es par

9 * 3 = 27 es impar

9 * 4 = 36 es par

9 * 5 = 45 es impar

FOR Con Parametros:

9 * 1 = 9 es impar

9 * 2 = 18 es par

9 * 3 = 27 es impar

9 * 4 = 36 es par

9 * 5 = 45 es impar

Nombre de la función: cicloWhileExp(pcont, pnum)		Versión: 2.0
Descripción: Función que muestra la tabla de multiplicar del 9 hasta el 5 y dice si el resultado es par o impar con WHILE		
Cont, num, mult, parImpar	Tipo de variable: Numérico	
Código: <pre>const cicloWhileExp = function (pcont, pnum) { let cont = pcont let num = pnum while (cont <= 5) { let mult = num * cont let parImpar = esParImpar(mult) console.log(num+" * "+cont+" = "+mult+" es "+parImpar) cont = cont + 1 } return "" }</pre>		

WHILE Con Expresion:

9 * 1 = 9 es impar

9 * 2 = 18 es par

9 * 3 = 27 es impar

9 * 4 = 36 es par

9 * 5 = 45 es impar

WHILE Con Expresion:

9 * 1 = 9 es impar

9 * 2 = 18 es par

9 * 3 = 27 es impar

9 * 4 = 36 es par

9 * 5 = 45 es impar



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: cicloForExp(pcont, pnum)		Versión: 2.0
Descripción: Función que muestra la tabla de multiplicar del 9 hasta el 5 y dice si el resultado es par o impar con FOR		
Cont, num, mult, parImpar	Tipo de variable: Numérico	
Código:		

```
const cicloForExp = function (pcont, pnum) {  
  let cont = pcont  
  let num = pnum  
  for (cont; cont <= 5; cont++) {  
    let mult = num * cont  
    let parImpar = esParImpar(mult)  
    console.log(num + " * " + cont + " = " + mult + " es " + parImpar)  
  }  
  return ""  
}
```

FOR Con Expresion:
9 * 1 = 9 es impar
9 * 2 = 18 es par
9 * 3 = 27 es impar
9 * 4 = 36 es par
9 * 5 = 45 es impar

FOR Con Expresion:

```
9 * 1 = 9 es impar  
9 * 2 = 18 es par  
9 * 3 = 27 es impar  
9 * 4 = 36 es par  
9 * 5 = 45 es impar
```

23-24.

Nombre de la función: esParImpar(num)		Versión: 1.0
Descripción: Función que dice si el numero dado es par(buzz) o impar(bass)		
num	Tipo de variable: Numérico	
Código: <pre>function esParImpar(num) { if (num % 2 === 0) { return "buzz" } else { return "bass" } }</pre>		



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: cicloWhile(ptab, pcont, ppar, pimpar) Versión: 1.0	
Descripción: Función que muestra la tabla de multiplicar del 1 hasta la del 5 e identifica los pares(buzz) e impares (bass) con WHILE	
Tab, cont, par, impar, mult, parImpar	Tipo de variable: Numérico
Código:	
<pre>function cicloWhile(ptab, pcont, ppar, pimpar) { let tab = ptab let cont = pcont let par = ppar let impar = pimpar while (tab < 5) { tab = tab + 1 cont = 1 while (cont <= 5) { let mult = tab * cont let parImpar = esParImpar(mult) if (mult % 2 == 0) { console.log(tab + " * " + cont + " = " + mult + parImpar) par = par + 1 } else { console.log(tab + " * " + cont + " = " + mult + parImpar) impar = impar + 1 } cont = cont + 1 } } return "Total Pares: " + par + "\n" + "Total Impares: " + impar }</pre>	
WHILE Con Parametros: 1 * 1 = 1 bass 1 * 2 = 2 buzz 1 * 3 = 3 bass 1 * 4 = 4 buzz 1 * 5 = 5 bass 2 * 1 = 2 buzz 2 * 2 = 4 buzz 2 * 3 = 6 buzz 2 * 4 = 8 buzz 2 * 5 = 10 buzz 3 * 1 = 3 bass 3 * 2 = 6 buzz 3 * 3 = 9 bass 3 * 4 = 12 buzz 3 * 5 = 15 bass 4 * 1 = 4 buzz 4 * 2 = 8 buzz 4 * 3 = 12 buzz 4 * 4 = 16 buzz 4 * 5 = 20 buzz 5 * 1 = 5 bass 5 * 2 = 10 buzz 5 * 3 = 15 bass 5 * 4 = 20 buzz 5 * 5 = 25 bass Total Pares: 16 Total Impares: 9	



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: cicloFor(ptab, pcont, ppar, pimpar)

Versión: 1.0

Descripción:

Función que muestra la tabla de multiplicar del 1 hasta la del 5 e identifica los pares(buzz) e impares (bass) con FOR

Tab, cont, par, impar,
mult, parImpar

Tipo de variable: Numérico

Código:

```
function cicloFor(ptab, pcont, ppar, pimpar) {  
  let tab = ptab  
  let cont = pcont  
  let par = ppar  
  let impar = pimpar  
  for (tab; tab <= 5; tab++) {  
    cont = 1  
    for (cont; cont <= 5; cont++) {  
      let mult = tab * cont  
      let parImpar = esParImpar(mult)  
      if (mult % 2 == 0) {  
        console.log(tab + " * " + cont + " = " + mult + parImpar)  
        par = par + 1  
      }  
      else {  
        console.log(tab + " * " + cont + " = " + mult + parImpar)  
        impar = impar + 1  
      }  
    }  
  }  
  return "Total Pares: " + par + "\n" + "Total Impares: " + impar  
}
```

FOR Con Parametros:

```
1 * 1 = 1 bass  
1 * 2 = 2 buzz  
1 * 3 = 3 bass  
1 * 4 = 4 buzz  
1 * 5 = 5 bass  
2 * 1 = 2 buzz  
2 * 2 = 4 buzz  
2 * 3 = 6 buzz  
2 * 4 = 8 buzz  
2 * 5 = 10 buzz  
3 * 1 = 3 bass  
3 * 2 = 6 buzz  
3 * 3 = 9 bass  
3 * 4 = 12 buzz  
3 * 5 = 15 bass  
4 * 1 = 4 buzz  
4 * 2 = 8 buzz  
4 * 3 = 12 buzz  
4 * 4 = 16 buzz  
4 * 5 = 20 buzz  
5 * 1 = 5 bass  
5 * 2 = 10 buzz  
5 * 3 = 15 bass  
5 * 4 = 20 buzz  
5 * 5 = 25 bass  
Total Pares: 16  
Total Impares: 9
```




Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: cicloWhileExp(ptab, pcont, ppar, pimpar)

Versión: 2.0

Descripción:

Función que muestra la tabla de multiplicar del 1 hasta la del 5 e identifica los pares(buzz) e impares (bass) con WHILE

Tab, cont, par, impar, mult, parImpar

Tipo de variable: Numérico

Código:

```
const cicloWhileExp = function (ptab, pcont, ppar, pimpar) {  
  ...let tab = ptab  
  ...let cont = pcont  
  ...let par = ppar  
  ...let impar = pimpar  
  ...while (tab < 5) {  
    ...tab = tab + 1  
    ...cont = 1  
    ...while (cont <= 5) {  
      ...let mult = tab * cont  
      ...let parImpar = esParImpar(mult)  
      ...if (mult % 2 == 0) {  
        ...console.log(tab + " * " + cont + " = " + mult + parImpar)  
        ...par = par + 1  
      } else {  
        ...console.log(tab + " * " + cont + " = " + mult + parImpar)  
        ...impar = impar + 1  
      }  
      ...cont = cont + 1  
    }  
  }  
  ...return "Total Pares: " + par + "\n" + "Total Impares: " + impar  
}
```

WHILE Con Expression:

```
1 * 1 = 1 bass  
1 * 2 = 2 buzz  
1 * 3 = 3 bass  
1 * 4 = 4 buzz  
1 * 5 = 5 bass  
2 * 1 = 2 buzz  
2 * 2 = 4 buzz  
2 * 3 = 6 buzz  
2 * 4 = 8 buzz  
2 * 5 = 10 buzz  
3 * 1 = 3 bass  
3 * 2 = 6 buzz  
3 * 3 = 9 bass  
3 * 4 = 12 buzz  
3 * 5 = 15 bass  
4 * 1 = 4 buzz  
4 * 2 = 8 buzz  
4 * 3 = 12 buzz  
4 * 4 = 16 buzz  
4 * 5 = 20 buzz  
5 * 1 = 5 bass  
5 * 2 = 10 buzz  
5 * 3 = 15 bass  
5 * 4 = 20 buzz  
5 * 5 = 25 bass  
Total Pares: 16  
Total Impares: 9
```



Tecnólogo en Análisis y Desarrollo de Software

Ficha

Funciones JS

Nombre de la función: cicloForExp(ptab, pcont, ppar, pimpar)

Versión: 2.0

Descripción:

Función que muestra la tabla de multiplicar del 1 hasta la del 5 e identifica los pares(buzz) e impares (bass) con FOR

Tab, cont, par, impar, mult, parImpar

Tipo de variable: Numérico

Código:

```
const cicloForExp = function (ptab, pcont, ppar, pimpar) {  
  let tab = ptab  
  let cont = pcont  
  let par = ppar  
  let impar = pimpar  
  for (tab; tab <= 5; tab++) {  
    cont = 1  
    for (cont; cont <= 5; cont++) {  
      let mult = tab * cont  
      let parImpar = esParImpar(mult)  
      if (mult % 2 == 0) {  
        console.log(tab + " * " + cont + " = " + mult + parImpar)  
        par = par + 1  
      }  
      else {  
        console.log(tab + " * " + cont + " = " + mult + parImpar)  
        impar = impar + 1  
      }  
    }  
  }  
  return "Total Pares: " + par + "\n" + "Total Impares: " + impar  
}
```

FOR Con Expresion:

```
1 * 1 = 1 bass  
1 * 2 = 2 buzz  
1 * 3 = 3 bass  
1 * 4 = 4 buzz  
1 * 5 = 5 bass  
2 * 1 = 2 buzz  
2 * 2 = 4 buzz  
2 * 3 = 6 buzz  
2 * 4 = 8 buzz  
2 * 5 = 10 buzz  
3 * 1 = 3 bass  
3 * 2 = 6 buzz  
3 * 3 = 9 bass  
3 * 4 = 12 buzz  
3 * 5 = 15 bass  
4 * 1 = 4 buzz  
4 * 2 = 8 buzz  
4 * 3 = 12 buzz  
4 * 4 = 16 buzz  
4 * 5 = 20 buzz  
5 * 1 = 5 bass  
5 * 2 = 10 buzz  
5 * 3 = 15 bass  
5 * 4 = 20 buzz  
5 * 5 = 25 bass  
Total Pares: 16  
Total Impares: 9
```