Tema:

Manual de Funciones

Instructor:

Andrés Moreno Collazos

Estudiante:

Jhampier Santos Ortiz

Neiva-Huila

2024

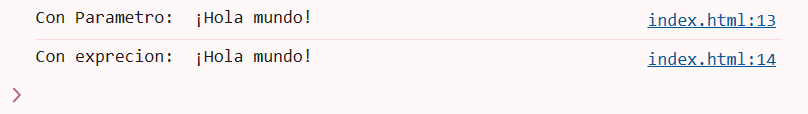
Ejercicios de JS normales

1.

| **Nombre de la función: saludo(*psaludo)*** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  **Imprimir “Hola Mundo” dentro de una variable.** | | |
| saludar | Tipo de variable:  saludar (string) | |
| Código:  function saludo(psaludo){ *//parametro*      let saludar = psaludo;      return saludar;  } | | |
|  | | |

| **Nombre de la función: saludoExp(psaludo)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  **Imprimir “Hola Mundo” dentro de una variable.** | | |
| saludar | Tipo de variable:  saludar (string) | |
| Código:  const saludoExp = function(psaludo){ *//expresion*      let saludar = psaludo;      return saludar;  } | | |

Prueba de ambos:

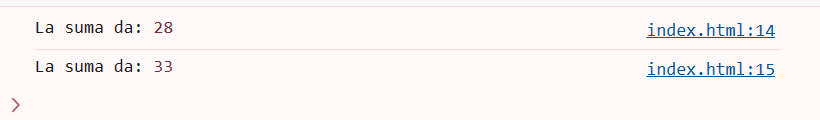


2.

| **Nombre de la función: suma(*pnumero1*, *pnumero2*)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  **Imprimir la suma de dos números.** | | |
| Numero1  numero2  sumar | Tipo de variable:  Numero1(int)  numero2(int)  sumar(int) | |
| Código:  function suma(pnumero1, pnumero2){ //parametro      let numero1 = pnumero1;      let numero2 = pnumero2;      let sumar;      sumar = numero1 + numero2      return sumar;  } | | |

| **Nombre de la función: sumarExp** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  **Imprimir la suma de dos números.** | | |
| sumar | Tipo de variable:  Numero1(int)  numero2(int)  sumar(int) | |
| Código:  const sumaExp = function(pnumero1, pnumero2){ //expresion      let numero1 = pnumero1;      let numero2 = pnumero2;      let sumar;      sumar = numero1 + numero2;      return sumar;  } | | |

Prueba de ambos:

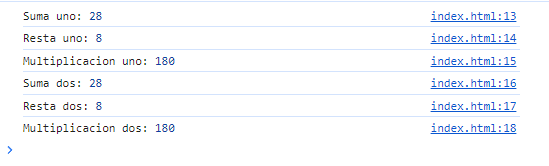


3.

| **Nombre de la función: operacion(*poperador*,*pnumero1*,*pnumero2*), multiplicacion(*pnumero1*,*pnumero2),* division(*pnumero1*,*pnumero2),***  **resta(*pnumero1*,*pnumero2)*** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  **imprimir la resta, la multiplicacion y la division de dos numeros.** | | |
| operador numero1  numero2  restar  multiplicar  dividor  operacionResult | Tipo de variable:  operador (string)  numero1(int)  numero2(int)  restar(int)  multiplicar(int)  dividor(float)  operacionResult(Alfanumerico) | |
| function resta(pnumero1,pnumero2){ // parametro     let restar;     numero1 = pnumero1;     numero2 = pnumero2;     restar = pnumero1 – pnumero2;     return restar;  }  function multiplicacion(pnumero1,pnumero2){     let multiplicar;     numero1 = pnumero1;     numero2 = pnumero2;     multiplicar = pnumero1 \* pnumero2;     return multiplicar;  }  function division(pnumero1,pnumero2){     let dividor;     numero1 = pnumero1;     numero1 = pnumero2;     dividor = pnumero1 / pnumero2;     return dividor;  }  function operacion(poperador,pnumero1,pnumero2){     let operador = poperador;     numero1 = pnumero1;     numero2 = pnumero2;      if (operador == "resta") {         operacionResult = resta(pnumero1,pnumero2);      }else if (operador == "multiplicacion") {         operacionResult = multiplicacion(pnumero1,pnumero2);      }else if (operador == "division") {         operacionResult = division(pnumero1,pnumero2);      }else{         operacionResult = "Error";      }      return operacionResult;  } | | |

| **Nombre de la función: operacionExp, restaExp, multiplicacionExp*,* divisionExp** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  **imprimir la resta, la multiplicacion y la division de dos numeros.** | | |
| operador numero1  numero2  restar  multiplicar  dividor  operacionResult | Tipo de variable:  operador (string)  pnumero1(int)  pnumero2(int)  restar(int)  multiplicar(int)  dividor(float)  operacionResult(Alfanumerico) | |
| const restaExp=function(pnumero1,pnumero2){ // expresion      let restar;      numero1 = pnumero1;      numero2 = pnumero2;      restar = pnumeroUno - pnumeroDos;      return restar;  }  const multiplicacionExp=function(pnumero1,pnumero2){      let multiplicar;      numero1 = pnumero1;      numero2 = pnumero2;      multiplicar = pnumero1 \* pnumero2;      return multiplicar;  }  const divisionExp=function(pnumero1,pnumero2){      let dividor;      numero1 = pnumero1;      numero2 = pnumero2;      dividor = pnumero1 / pnumero2;      return dividor;  }  const operacionExp = function(poperador,pnumeroUno,pnumeroDos){      let operador = poperador;      numero1 = pnumero1;      numero2 = pnumero2;       if (operador == "resta") {          operacionResult = restaExp(pnumero1,pnumero2);       }else if (operador == "multiplicacion") {          operacionResult = multiplicacionExp(pnumero1,pnumero2);       }else if (operador == "division") {          operacionResult = divisionExp(pnumero1,pnumero2);       }else{          operacionResult = "Error";       }       return operacionResult;  } | | |

Prueba de ambos:



4.

| **Nombre de la función: porcentaje(p*num1*)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  **Imprimir el porcentaje de un número.** | | |
| Porcen  num1 | Tipo de variable:  porcen(float)  num1(int) | |
| Código:  function porcentaje(pnum1){ //parametro      let porcen;      let num1 = pnum1;      porcen = num1 / 100;      return porcen;  } | | |

| **Nombre de la función: porcentajeExp** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  **Imprimir el porcentaje de un número.** | | |
| porcen  num | Tipo de variable:  porcen (float)  num1 (int) | |
| Código:  const porcentajeExp = function (pnum1){ //expresion      let porcen;      let num1 = pnum1;      porcen = num1 / 100;      return porcen;  } | | |

Prueba de ambos:

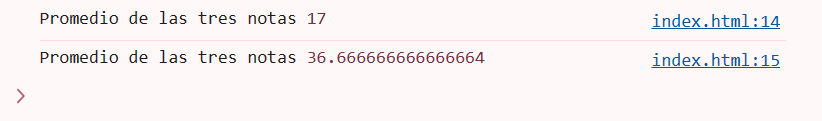


5.

| **Nombre de la función: promedio(p*not1*,p*not2*,p*not3*)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Imprimir el promedio de 3 notas. | | |
| not1  not2  not3  prom | Tipo de variable:  not1(int)  not2(int)  not3(int)  prom(float) | |
| Código:  function promedio(pnot1, pnot2, pnot3){ // parametro          let prom;          let not1 = pnot1;          let not2 = pnot2;          let not3 = pnot3;          prom = (not1+not2+not3)/3;          return prom;  } | | |

| **Nombre de la función: promediosExp** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Imprimir el promedio de 3 notas. | | |
| not1  not2  not3  prom | Tipo de variable:  not1(int)  not2(int)  not3(int)  prom(float) | |
| Código:  const promedioExp = function(pnot1, pnot2, pnot3){ // expresion          let prom;          let not1 = pnot1;          let not2 = pnot2;          let not3 = pnot3;          prom = (not1+not2+not3)/3;          return prom;  } | | |

Prueba de ambos:

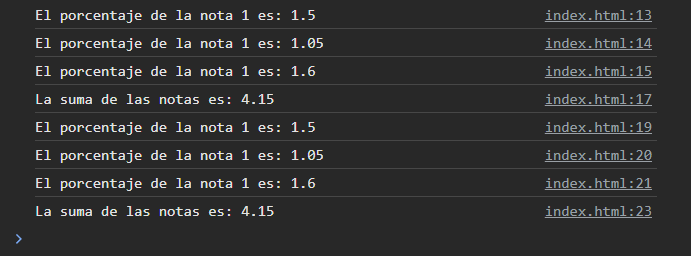


6.

| **Nombre de la función: sumar(pnota,pporcenNota)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  a. El porcentaje de 3 notas, la nota 1 =30%, la nota2=30% y la nota3 =40%.  b. Sumar el resultado de los porcentajes de las notas. | | |
| Sum  nota1  nota2  nota3 | Tipo de variable:  sum(float)  nota1(int)  nota2(int)  nota3(int) | |
| Código:  function sumar(pnota, pporcenNota){      let sum;      let nota = pnota;      let porcenNota = pporcenNota;      suma = nota \* porcenNota      return sum;  }  nota1 = sumar(5,0.3);  nota2 = sumar(3.5,0.3);  nota3 = sumar(4,0.4);  let sum = nota1+nota2+nota3 | | |

| **Nombre de la función: sumarExp** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  a. El porcentaje de 3 notas, la nota 1 =30%, la nota2=30% y la nota3 =40%.  b. Sumar el resultado de los porcentajes de las notas. | | |
| porcenNota  sum2  nota4  nota5  nota6 | Tipo de variable:  porcenNota(float)  sum2float)  nota4(int)  nota5(int)  nota6(int) | |
| Código:  const sumarExp = function(pnota, pporcenNota){      let sum2;      let nota = pnota;      let porcenNota = pporcenNota;      sum2 = nota \* porcenNota      return sum2;  }  nota4 = sumar(5,0.3);  nota5 = sumar(3.5,0.3);  nota6 = sumar(4,0.4);  let sum2 = nota4+nota5+nota6 | | |

Prueba de ambos:

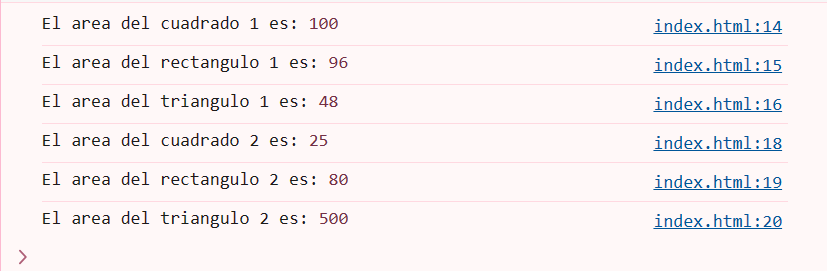


7.

| **Nombre de la función: áreas(operador,pbase,paltura,plad)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  realizar un diagrama de flujo que imprima las áreas de las figuras geométricas.  Teniendo en cuanta:  Cuadrado = lado x lado.  Rectángulo= base x altura  Triangulo = (base x altura) /2 | | |
| altura  lad  area  operador  base | Tipo de variable:  areas(Alfanumerica)  operador(Alfanumerica)  base(int)  altura(int)  lad(int) | |
| function areas(operador,pbase, paltura,plad ){ // parametro      let area      if (operador=="rectangulo") {          area = pbase\*paltura      }else if(operador=="triangulo"){          area = (pbase\*paltura)/2      }      else if(operador=="cuadrado"){          area = plad\*plad      }else{          area ="Error"}          return area  } | | |

| **Nombre de la función: areasExp(operador,pbase,paltura,plad)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  realizar un diagrama de flujo que imprima las áreas de las figuras geométricas.  Teniendo en cuanta:  Cuadrado = lado x lado.  Rectángulo= base x altura  Triangulo = (base x altura) /2 | | |
| base  altura  lad  area  operador | Tipo de variable:  areas(Alfanumerica)  operador(Alfanumerica)  base(int)  altura(int)  lad(int) | |
| const areasExp = function(operador,pbase, paltura,plad ){ // expresion      let area      if (operador=="rectangulo") {          area = pbase\*paltura      }else if(operador=="triangulo"){          area = (pbase\*paltura)/2      }      else if(operador=="cuadrado"){          area = plad\*plad      }else{          area ="Error"}          return area  } | | |

Prueba de ambos:

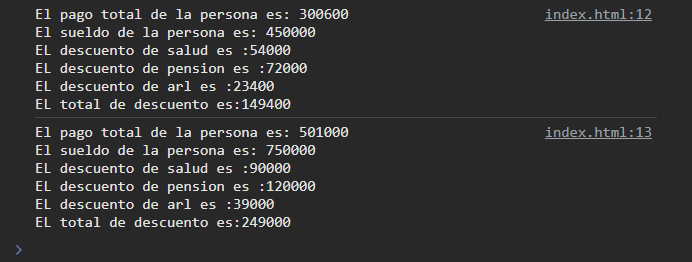


8.

| **Nombre de la función: pagPerson(valorDia,diasTrabajados)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  un diagrama de flujo que imprima el pago total de una persona, sabiendo que el sueldo es igual a los días por el valor de los días, imprimir salud, pensión, y arl sabiendo que la suma de la salud, pensión y arl se descuentan del sueldo de la persona. | | |
| sueldo  pensión  descuento  pagTotal  arl  salud | Tipo de variable:  pensión(int)  descuento(int)  pagTotal(int)  sueldo(int)  arl(int)  salud(int) | |
| function pagPerson(valorDia, diasTrabajados) { // parametro      let sueldo      let arl      let salud      let pension      let descuentos      let pagTotal      sueldo = valorDia\* diasTrabajados      salud= sueldo \* 0.12      pension = sueldo \*0.16      arl =sueldo \* 0.052      descuento = salud+ pension+ arl      pagoTotal= sueldo - descuento  return [pagTotal+" \n"+"El sueldo de la persona es: "+sueldo+" \n"+          "EL descuento de salud es :"+salud+" \n"+"EL descuento de pension es :"+pension+" \n"+      "EL descuento de arl es :"+arl+" \n"+"EL total de descuento es:"+descuento]  } | | |

| **Nombre de la función: pagPersonexp(valorDia,diasTrabajados)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  un diagrama de flujo que imprima el pago total de una persona, sabiendo que el sueldo es igual a los días por el valor de los días, imprimir salud, pensión, y arl sabiendo que la suma de la salud, pensión y arl se descuentan del sueldo de la persona. | | |
| sueldo  arl  salud  pensión  descuento  pagTotal | Tipo de variable:  sueldo(int)  arl(int)  salud(int)  pensión(int)  descuento(int)  pagTotal(int) | |
| const pagPersonexp = function(valorDia, diasTrabajados) { // expresion      let sueldo      let arl      let salud      let pension      let descuento      let pagTotal      sueldo = valorDia\* diasTrabajados      salud= sueldo \* 0.12      pension = sueldo \*0.16      arl =sueldo \* 0.052      descuento = salud+ pension+ arl      pagoTotal= sueldo - descuento  return [pagTotal+" \n"+"El sueldo de la persona es: "+sueldo+" \n"+          "EL descuento de salud es :"+salud+" \n"+"EL descuento de pension es :"+pension+" \n"+      "EL descuento de arl es :"+arl+" \n"+"EL total de descuento es:"+descuento]  } | | |

Prueba de ambos:



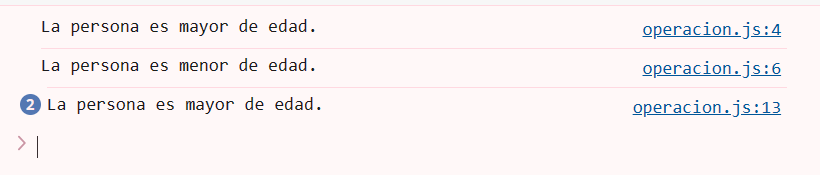
Ejercicios en JS de condicionales

1.

| **Nombre de la función: verificarEdad(pedad)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Imprimir si es mayor o menor de edad. | | |
| Edad | Tipo de variable:  edad(int) | |
| Código:  function verificarEdad(edad) {      let verificarEdadior      if (edad >= 18) {          console.log("La persona es mayor de edad.");      } else {          console.log("La persona es menor de edad.");      }      return(verificarEdadior)} | | |

| **Nombre de la función: verificarEdadexp(pedad)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Imprimir si es mayor o menor de edad. | | |
| Edad | Tipo de variable:  edad(int) | |
| Código:  cconst verificarEdadexp=function(edad) {      let verificarEdadior      if (edad >= 18) {          console.log("La persona es mayor de edad.");      } else {          console.log("La persona es menor de edad.");      }      return(verificarEdadior)} | | |

Prueba de ambos:

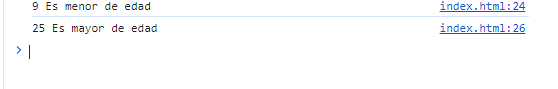


2.

| **Nombre de la función: edades(p*anioNac,*p*anioAct*)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Calcular la edad de una persona, imprimirla edad y si es mayor o menor de edad. | | |
| edad | Tipo de variable:  edad(int)  panioAct(int)  panioNac(int) | |
| Código:  function edades(panioNac,panioAct){;      edad = panioAct - panioNac;      if(edad>17){          edad = edad + " Es mayor de edad "      }else{          edad = edad + " Es menor de edad "}      return edad; } | | |

| **Nombre de la función: edadesExp(p*anioNac,*p*anioAct*)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Calcular la edad de una persona, imprimirla edad y si es mayor o menor de edad. | | |
| edad | Tipo de variable:  edad(int)  panioAct(int)  panioNac(int) | |
| Código:  const edadesExp = function(panioNac,panioAct){;      edad = panioAct - panioNac;      if(edad>17){          edad = edad + " Es mayor de edad "      else{          edad = edad + " Es menor de edad " }      return edad; } | | |

Prueba de ambos:



3.

| **Nombre de la función: numeros(pnum1,pnum2)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Imprimir el número mayor, el menor o si son iguales los dos. | | |
| numero  num1  num2 | Tipo de variable:  num2(Int)  num1(Int)  numer(string) | |
| Código:  function numeros(pnum1,pnum2){      let numer;      let num1 = pnum1;      let num2 = pnum2;      if(num1==num2){          numer = "Son iguales"      } else if(num1>num2){          numer = "Numero 1 es mayor"      } else{          numer = "Numero 2 es mayor"      }      return numer;  } | | |

| **Nombre de la función: numerosExp(pnum1,pnum2)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Imprimir el número mayor, el menor o si son iguales los dos. | | |
| numero  num1  num2 | Tipo de variable:  Num2(Int)  num1(Int)  numer(string) | |
| const numerosExp = function(pnum1,pnum2){      let numer;      let num1 = pnum1;      let num2 = pnum2;      if(num1==num2){          numer = "Son iguales"      } else if(num1>num2){          numer = "Numero 1 es mayor"      } else{          numer = "Numero 2 es mayor"      }      return numer;  } | | |

Prueba de ambos:



4.

| **Nombre de la función: area1(pladCuad1,pladCuad2,pladCuad3)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Calcular el área de 3 cuadrado e imprimir si las áreas son iguales o mayor. | | |
| pladCuad1  pladCuad2  pladCuad3  cuad1  cuad2  cuad3 | Tipo de variable:  pladCuad1(int)  pladCuad2 (int)  pladCuad3(int)  cuad1(int)  cuad2(int)  cuad3(int) | |
| function area1(pladCuad1,pladCuad2,pladCuad3){ *//parametro*      let cuad1 = pladCuad1;      let cuad2 = pladCuad2;      let cuad3 = pladCuad3;      cuad1=pladCuad1\* pladCuad1      cuad2=pladCuad2\* pladCuad2      cuad3=pladCuad3\* pladCuad3      if(cuad1==cuad2 && cuad1==cuad3){          return ("Las areas de los cuadrados son iguales")      }      else{          if(cuad1>cuad2 && cuad1>cuad3){              return ("El area del cuadrado 1 es mayor")          }          else{              if(cuad2>cuad3 && cuad2>cuad1){                  return("El area del cuadrado 2 es mayor")              }              else{                  return("El area del cuadrado 3 es mayor")              }          }        }  } | | |

| **Nombre de la función: area2Exp(pladCuad1,pladCuad2,pladCuad3)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Calcular el área de 3 cuadrado e imprimir si las áreas son iguales o mayor. | | |
| pladCuad1  pladCuad2  pladCuad3  cuad1  cuad2  cuad3 | Tipo de variable:  pladCuad1(int)  pladCuad2 (int)  pladCuad3(int)  cuad1(int)  cuad2(int)  cuad3(int) | |
| const area2Exp = function(pladCuad1,pladCuad2,pladCuad3){ *//expresion*      let cuad1 = pladCuad1;      let cuad2 = pladCuad2;      let cuad3 = pladCuad3;      cuad1=pladCuad1\* pladCuad1      cuad2=pladCuad2\* pladCuad2      cuad3=pladCuad3\* pladCuad3      if(cuad1==cuad2 && cuad1==cuad3){          return ("Las areas de los cuadrados son iguales")      }      else{          if(cuad1>cuad2 && cuad1>cuad3){              return ("El area del cuadrado 1 es mayor")          }          else{              if(cuad2>cuad3 && cuad2>cuad1){                  return("El area del cuadrado 2 es mayor")              }              else{                  return("El area del cuadrado 3 es mayor")              }          }      } | | |

Prueba de ambos:

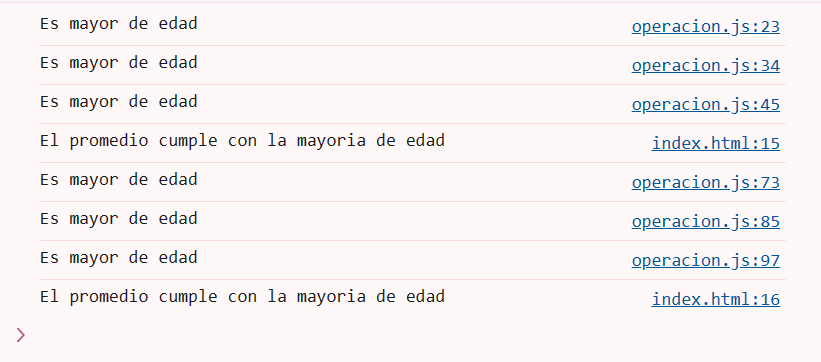


5.

| **Nombre de la función: edades(*anionac1*,*anionac2*,*anionac3*),**  **anios(*anionac1*), anios2(*anionac2*),**  **anios3(*anionac3*)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Realizar un diagrama de flujo que calcule la edad de 3 personas e imprimir si cada persona es mayor de edad, calcular e imprimir el mismo diagrama de promedio de las 3 edades y si el promedio de edades si es mayor de edad. | | |
| aninac1  aninac2  aninac3  edad1  edad2  edad3  eda1  eda2  eda3  promedio  prom | Tipo de variable:  Aninac2(int)  Aninac3(int)  Aninac4(int)  edad2(int)  edad3(int)  edad4(int)  eda2(string)  eda3(string)  eda4(string)  promedio(int)  prom(string) | |
| function edades(anionac1,anionac2,anionac3) {      let aninac1=anionac1;      let aninac2=anionac2;      let aninac3=anionac3;      let promedio=anios(anionac1)+anios2(anionac2)+anios3(anionac3);      let prom      if (promedio>17) {          prom="El promedio cumple con la mayoria de edad";      } else {         prom= "El promedio no comple con la mayoria de edad";      }      return prom  }  function anios(anionac1,) {      let aninac1=anionac1;      let edad1 = 2024-aninac1;      if (edad1>17) {          eda1 ="Es mayor de edad";      } else{          eda1="Es menor de edad";      }      console.log(eda1);      return edad1  }  function anios2(anionac2,) {      let aninac2=anionac2;      let edad2 = 2024-aninac2;      if (edad2>17) {          eda2 ="Es mayor de edad";      } else{          eda2="Es menor de edad";      }      console.log(eda2);      return edad2  }  function anios3(anionac3,) {      let aninac3=anionac3;      let edad3 = 2024-aninac3;      if (edad3>17) {          eda3 ="Es mayor de edad";      } else{          eda3="Es menor de edad";      }      console.log(eda3);      return edad3  } | | |

| **Nombre de la función: edadesExp,**  **anios4, anios5, anios6** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Realizar un diagrama de flujo que calcule la edad de 3 personas e imprimir si cada persona es mayor de edad, calcular e imprimir el mismo diagrama de promedio de las 3 edades y si el promedio de edades si es mayor de edad. | | |
| aninac1  aninac2  aninac3  edad1  edad2  edad3  eda1  eda2  eda3  promedio  prom | Tipo de variable:  aninac4(int)  aninac5(int)  aninac6(int)  edad4(int)  edad5(int)  edad6(int)  eda4(string)  eda5(string)  eda6(string)  promedio(int)  prom(string) | |
| const edadesExp=function(anionac1,anionac2,anionac3) {      let aninac1=anionac1;      let aninac2=anionac2;      let aninac3=anionac3;        let promedio=anios4(anionac1)+anios5(anionac2)+anios6(anionac3);      let prom      if (promedio>17) {          prom="El promedio cumple con la mayoria de edad";      } else {         prom= "El promedio no comple con la mayoria de edad";      }      return prom  }  const anios4=function(anionac4,) {      let aninac4=anionac4;      let edad4 = 2024-aninac4;      let eda4;      if (edad4>17) {          eda4 ="Es mayor de edad";      } else{          eda4="Es menor de edad";      }      console.log(eda4);      return edad4;  }  const anios5=function(anionac5,) {      let aninac5=anionac5;      let edad5 = 2024-aninac5;      let eda5;      if (edad5>17) {          eda5 ="Es mayor de edad";      } else{          eda5="Es menor de edad";      }      console.log(eda5);      return edad5  }  const anios6=function(anionac6,) {      let aninac6=anionac6;      let edad6 = 2024-aninac6;      let eda6;      if (edad6>17) {          eda6 ="Es mayor de edad";      } else{          eda6="Es menor de edad";      }      console.log(eda6);      return edad6;  } | | |

Prueba de ambos:

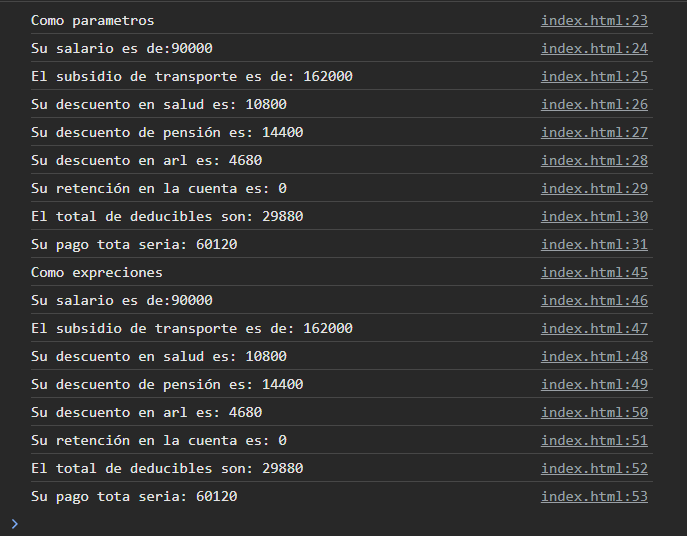


6.

| **Nombre de la función: salarioExp(pDiasTrabajados,pValorDia) subTransExp(pDiasTrabajados,pValorDia) saludExp(pDiasTrabajados,pValorDia)**  **retencionExp(pDiasTrabajados,pValorDia)**  **deducionesExp(pDiasTrabajados,pValorDia)**  **pagoFinalExp(pDiasTrabajados,pValorDia)**  **pensionExp(pDiasTrabajados,pValorDia)**  **arlExp(pDiasTrabajados,pValorDia)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Calcule el pago total del sueldo de una persona donde calcular e imprima lo siguiente:  a. salario de la persona.  b. si la persona gana más de dos salarios mínimos se suma a su sueldo, el subsidio de transporte de lo contrario sumara.  c. Calcular la salud, pensión y arl sabiendo que:  salud = salario x 0,12  pensión= salario x 0,16  arl = salario x 0,052  d. Si la persona gana mas de 4 salarios mínimos debe hacer una retención del 0,04 de un salario.  e. sumar los deducibles que son salud, pensión y arl, y restar el deducible al salario de la persona.  f. calcular e imprimir el total de pagar | | |
| diasTrabajados  valorDia  salarioPersona  salarioMininmo  pagSalud  pagoPension  pagArl  pagoDeducibl  pagoFinal  salarioTrans  subTransporte | Tipo de variable:  diasTrabajados(int)  valorDia(int)  salarioPersona(int)  pagSalud(float)  pagoPension(float)  pagArl(float)  pagoDeducible(int)  pagFina(int)  salarioMininmo(int)  salarioTrans(int)  subTransporte(int) | |
| function salarioExp (pDiasTrabajados,pValorDia) {      let salarioPersona      salarioPersona = pDiasTrabajados \* pValorDia      return  salarioPersona  }  function subTransExp (pDiasTrabajados,pValorDia) {      let salarioMinimo = 1600000      let salarioTrans = salarioExp (pDiasTrabajados,pValorDia)      let subTransporte      if (salarioTrans<=2\*salarioMinimo) {          subTransporte = 162000      }      else{          subTransporte = 0      }      return subTransporte  }  function saludExp (pDiasTrabajados,pValorDia){      let pagSalud      pagSalud = salarioExp(pDiasTrabajados,pValorDia) \* 0.12      return pagSalud}  function pensionExp(pDiasTrabajados,pValorDia){      let pagoPension      pagoPension =salarioExp(pDiasTrabajados,pValorDia) \* 0.16      return pagoPension}  function arlExp(pDiasTrabajados,pValorDia){      let pagArl      pagArl = salarioExp(pDiasTrabajados,pValorDia) \* 0.052      return pagArl  }  function retencionExp(pDiasTrabajados,pValorDia){      let reten      let salarioMinimo = 1600000      reten = salarioExp(pDiasTrabajados,pValorDia)      if (reten>4\*salarioMinimo) {          descuento = reten \* 0.04      }      else{          descuento = 0}      return descuento}  function deducionesExp (pDiasTrabajados,pValorDia){      let descuentos      descuentos = saludExp(pDiasTrabajados,pValorDia) + pensionExp(pDiasTrabajados,pValorDia) + arlExp(pDiasTrabajados,pValorDia) + retencionExp(pDiasTrabajados,pValorDia)      return descuentos}  function pagoFinalExp(pDiasTrabajados,pValorDia){      let pagTotal      pagTotal= salarioExp(pDiasTrabajados,pValorDia) - deducionesExp(pDiasTrabajados,pValorDia)      return pagTotal  } | | |

| **Nombre de la función: salarioExp(pDiasTrabajados,pValorDia) subTransExp(pDiasTrabajados,pValorDia) saludExp(pDiasTrabajados,pValorDia)**  **pensionExp(pDiasTrabajados,pValorDia)**  **arlExp(pDiasTrabajados,pValorDia)**  **retencionExp(pDiasTrabajados,pValorDia)**  **deducionesExp(pDiasTrabajados,pValorDia)**  **pagoFinalExp(pDiasTrabajados,pValorDia)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Calcule el pago total del sueldo de una persona donde calcular e imprima lo siguiente:  a. salario de la persona.  b. si la persona gana más de dos salarios mínimos se suma a su sueldo, el subsidio de transporte de lo contrario sumara.  c. Calcular la salud, pensión y arl sabiendo que:  salud = salario x 0,12  pensión= salario x 0,16  arl = salario x 0,052  d. Si la persona gana mas de 4 salarios mínimos debe hacer una retención del 0,04 de un salario.  e. sumar los deducibles que son salud, pensión y arl, y restar el deducible al salario de la persona.  f. calcular e imprimir el total de pagar | | |
| diasTrabajados  valorDia  salarioPersona  salarioMininmo  salarioTrans  subTransporte  pagSalud  pagPension  pagArl  pagoDeducibl  pagoFinal | Tipo de variable:  diasTrabajados(int)  valorDia(int)  salarioPersona(int)  salarioMininmo(int)  salarioTrans(int)  subTransporte(int)  pagSalud(float)  pagPension(float)  pagArl(float)  pagoDeducible(int)  pagFinal(int) | |
| const salud = function (pDiasTrabajados,pValorDia){      let pagSalud      pagSalud = salario(pDiasTrabajados,pValorDia) \* 0.12      return pagSalud}  const pension = function(pDiasTrabajados,pValorDia){      let pagPension      pagPension =salario(pDiasTrabajados,pValorDia) \* 0.16      return pagPension}  const arl = function(pDiasTrabajados,pValorDia){      let pagArl      pagArl = salario(pDiasTrabajados,pValorDia) \* 0.052      return pagArl  }  const retencion = function(pDiasTrabajados,pValorDia){      let reten      let salarioMinimo = 1600000      reten = salario(pDiasTrabajados,pValorDia)      if (reten>4\*salarioMinimo) {          descuento = reten \* 0.04      }      else{          descuento = 0}      return descuento}  const deduciones = function (pDiasTrabajados,pValorDia){      let descuentos      descuentos = salud(pDiasTrabajados,pValorDia) + pension(pDiasTrabajados,pValorDia) + arl(pDiasTrabajados,pValorDia) + retencion(pDiasTrabajados,pValorDia)      return descuentos  }  const pagoFinal = function(pDiasTrabajados,pValorDia){      let pagTotal      pagTotal= salario(pDiasTrabajados,pValorDia) - deduciones(pDiasTrabajados,pValorDia)      return pagTotal  }  function salarioExp (pDiasTrabajados,pValorDia) {      let salarioPersona      salarioPersona = pDiasTrabajados \* pValorDia      return  salarioPersona  }  function subTransExp (pDiasTrabajados,pValorDia) {      let salarioMinimo = 1600000      let salarioTrans = salarioExp (pDiasTrabajados,pValorDia)      let subTransporte      if (salarioTrans<=2\*salarioMinimo) {          subTransporte = 162000      }      else{          subTransporte = 0      }        return subTransporte  }  function saludExp (pDiasTrabajados,pValorDia){      let pagSalud      pagSalud = salarioExp(pDiasTrabajados,pValorDia) \* 0.12      return pagSalud  }  function pensionExp(pDiasTrabajados,pValorDia){      let pagoPension      pagoPension =salarioExp(pDiasTrabajados,pValorDia) \* 0.16      return pagoPension  }  function arlExp(pDiasTrabajados,pValorDia){      let pagArl      pagArl = salarioExp(pDiasTrabajados,pValorDia) \* 0.052      return pagArl  }  function retencionExp(pDiasTrabajados,pValorDia){      let reten      let salarioMinimo = 1600000      reten = salarioExp(pDiasTrabajados,pValorDia)      if (reten>4\*salarioMinimo) {          descuento = reten \* 0.04}      else{          descuento = 0}      return descuento}  function deducionesExp (pDiasTrabajados,pValorDia){      let descuentos      descuentos = saludExp(pDiasTrabajados,pValorDia) + pensionExp(pDiasTrabajados,pValorDia) + arlExp(pDiasTrabajados,pValorDia) + retencionExp(pDiasTrabajados,pValorDia)      return descuentos  }  function pagoFinalExp(pDiasTrabajados,pValorDia){      let pagoTotal      pagoTotal= salarioExp(pDiasTrabajados,pValorDia) - deducionesExp(pDiasTrabajados,pValorDia)      return pagTotal  } | | |

Prueba de ambos:

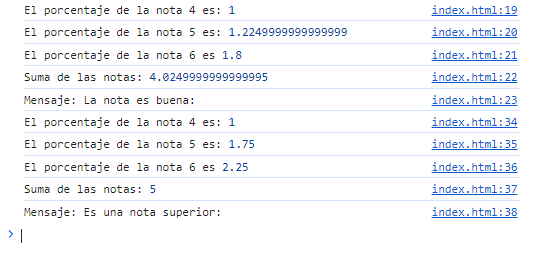


7.

| **Nombre de la función: nota1(nota), nota2(nota), nota3(nota), suma(sumaNotas)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Calcular tres notas e imprimir lo siguiente a. El 20 por ciento de la nota1 b. El 35 por ciento de In nota2 C. El 45 por ciento de la nota3 d. Sumar los porcentajes de las tres notas e imprimir los siguiente  Si la suma del porcentaje es mayor a 4.5 es una nota superior Si la suma del porcentaje esta entre 4.5 y 3.5 la nota es buena  Si la suma del porcentaje esta entre 3.5 y 3 la nota es media  Si la suma de porcentaje es menor de 3 es una nota mala. | | |
| calificacion  porcenNota  sumaNotas  nota | Tipo de variable:  califiaccion(int)  porcenNota(float)  suma(float)  nota(Alfanumerico) | |
| function nota1(nota) {      let calificacion      calificacion = nota      porceNota= calificacion \* 0.2        return porceNota    }  function nota2(nota) {       let calificacion      calificacion = nota   porceNota = calificacion \* 0.35   return porceNota  }  function nota3(nota) {      let calificacion      calificacion = nota      porceNota= calificacion \* 0.45      return porceNota  }  function suma(sumaNotas) {      let calificacion      calificación = sumaNotas      if (calificacion>=4.5) {          nota = "Es una nota superior: "      }      else{          if (calificacion<=4.5 & calificacion>=3.5) {              nota = "La nota es buena: "          }          else{              if(calificacion<=3.5 & calificacion>=3){                  nota = "La nota es media: "              }              else{                  nota = "Es una nota mala: "              }          }      }      return nota  } | | |

| **Nombre de la función: nota1(nota), nota2(nota), nota3(nota), suma(sumaNotas)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Calcular tres notas e imprimir lo siguiente a. El 20 por ciento de la nota1 b. El 35 por ciento de In nota2 C. El 45 por ciento de la nota3 d. Sumar los porcentajes de las tres notas e imprimir los siguiente  Si la suma del porcentaje es mayor a 4.5 es una nota superior Si la suma del porcentaje esta entre 4.5 y 3.5 la nota es buena  Si la suma del porcentaje esta entre 3.5 y 3 la nota es media  Si la suma de porcentaje es menor de 3 es una nota mala. | | |
| calificacion  porcenNota  sumaNotas  nota | Tipo de variable:  califiaccion(int)  porcenNota(float)  suma(float)  nota(Alfanumerico) | |
| const nota4 = function(nota) {      let calificacion      calificacion = nota      porceNota= calificacion \* 0.2        return porceNota    }  const nota5 = function(nota) {       let calificacion      calificacion = nota   porceNota= calificacion \* 0.35   return porceNota  }  const nota6 = function(nota) {      let calificacion      calificacion = nota      porceNota= calificacion \* 0.45      return porceNota  }  const suma2 = function(sumaNotas) {      let calificacion      calificacion= sumaNotas      if (calificacion>=4.5) {          nota = "Es una nota superior: "      }      else{          if (calificacion<=4.5 & calificacion>=3.5) {              nota = "La nota es buena: "          }          else{              if(calificacion<=3.5 & calificacion>=3){                  nota = "La nota es media: "              }              else{                  nota = "Es una nota mala: "              }          }      }      return nota  } | | |

Prueba de ambos:



Ejercicios en JS de ciclos

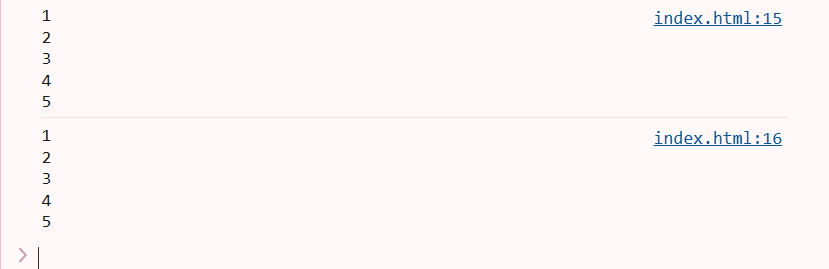
1.

While - mientras:

| **Nombre de la función: contar(pnumero)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Imprimir los números de 1 a 5. | | |
| contar  numero | Tipo de variable:  contar(int)  numero(int) | |
| Código:  function contar(pnumero) { *//parametro*      let numero = pnumero;      let contador = 1;      let resultadoContar = "";      while (contador <= numero) {          resultadoContar += contador + "\n";          contador = contador + 1;      }      return resultadoContar;  } | | |

| **Nombre de la función: contarExp(pnumero)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Imprimir los números de 1 a 5. | | |
| Contar  Numero | Tipo de variable:  contar(int)  numero(int) | |
| const contarExp = function(pnumero) { *//expresion*      let numero = pnumero;      let contador = 1;      let resultadoContar = "";      while (contador <= numero) {          resultadoContar += contador + "\n";          contador = contador + 1;      }      return resultadoContar;  } | | |

Prueba de ambos:

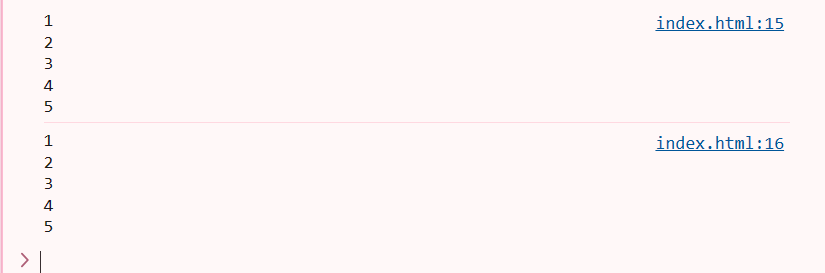


For - para:

| **Nombre de la función: contar(pnumero)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Imprimir los números de 1 a 5. | | |
| contar  numero | Tipo de variable:  contar(int)  numero(int) | |
| Código:  f function contar(pnumero) { *//parametro*      let contador      let numero=pnumero      let resultadoContar = ""        for(contador=1; contador<=numero; contador++){      resultadoContar+=contador + "\n"      }      return resultadoContar  } | | |

| **Nombre de la función: contarExp(pnumero)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Imprimir los números de 1 a 5. | | |
| contar  numero | Tipo de variable:  contar(int)  numero(int) | |
| const contarExp=function(pnumero) { *//expresion*      let contador      let numero=pnumero      let resultadoContar = ""        for(contador=1; contador<=numero; contador++){      resultadoContar+=contador + "\n"      }      return resultadoContar  } | | |

Prueba de ambas:



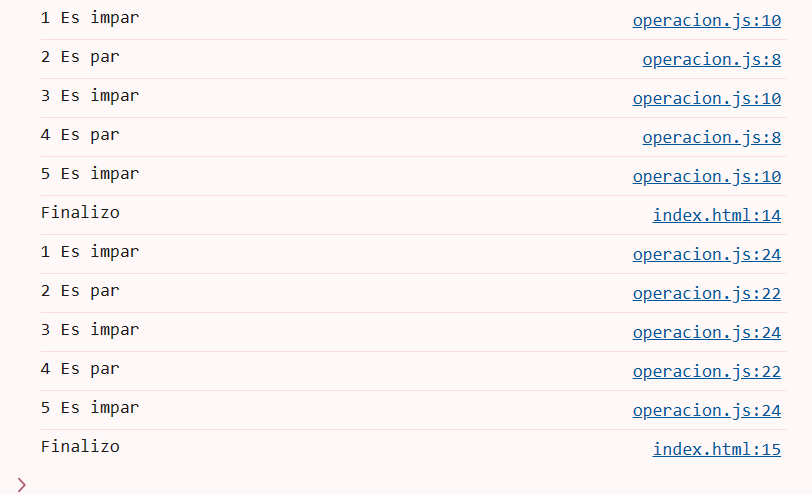
2.

While - mientras:

| **Nombre de la función: conteo(*contando*)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Contar de 1 a 5 e imprimir pares e impares. | | |
| Contar  Numero | Tipo de variable:  contar(int)  numero(int) | |
| Código:  function conteo(contando) {      let contar=contando;      let numero=4;      while (contar<=numero) {          contar++          if (contar%2==0) {              console.log(contar+" Es par");          } else {              console.log(contar+" Es impar");          }      }      return "Finalizo"  } | | |

| **Nombre de la función: conteoExp(*contando*)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Contar de 1 a 5 e imprimir pares e impares. | | |
| Contar  Numero | Tipo de variable:  contar(int)  numero(int) | |
| const conteoExp = function(contando) {      let contar=contando;      let numero=4;      while (contar<=numero) {          contar++          if (contar%2==0) {              console.log(contar+" Es par");          } else {              console.log(contar+" Es impar");          }      }      return "Finalizo"  } | | |

Prueba de ambas:

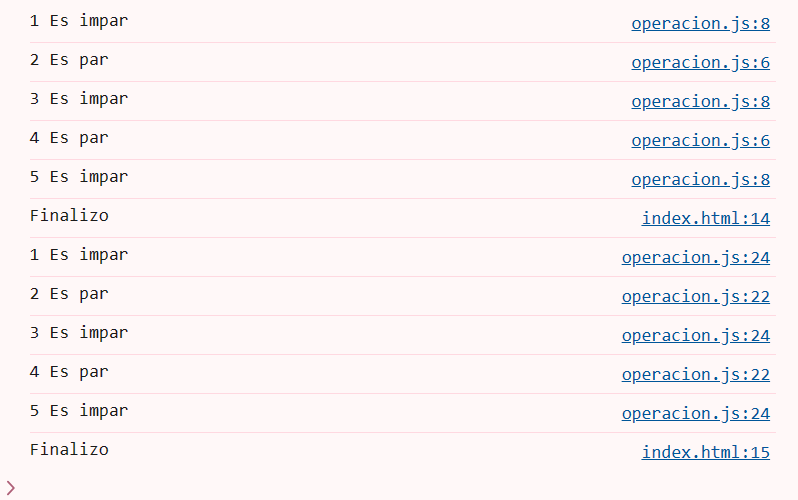


For - para:

| **Nombre de la función: conteo(*contando*)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Contar de 1 a 5 e imprimir pares e impares. | | |
| contar  numero | Tipo de variable:  contar(int)  numero(int) | |
| function conteo(contando){ // parametro      let contar=contando;      let numero=5;      for (contar=1; contar<=numero; contar++) {          if (contar%2==0) {              console.log(contar+" Es par");          } else {              console.log(contar+" Es impar");          }      }      return "Finalizo"  } | | |

| **Nombre de la función: conteoExp(contando)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Contar de 1 a 5 e imprimir pares e impares. | | |
| contar  numero | Tipo de variable:  contar(int)  numero(int) | |
| Código:  const conteoExp = function(contando){ //expresion      let contar=contando;      let numero=5;      for (contar=1; contar<=numero; contar++) {          if (contar%2==0) {              console.log(contar+" Es par");          } else {              console.log(contar+" Es impar");          }      }      return "Finalizo"  } | | |

Prueba de ambas:



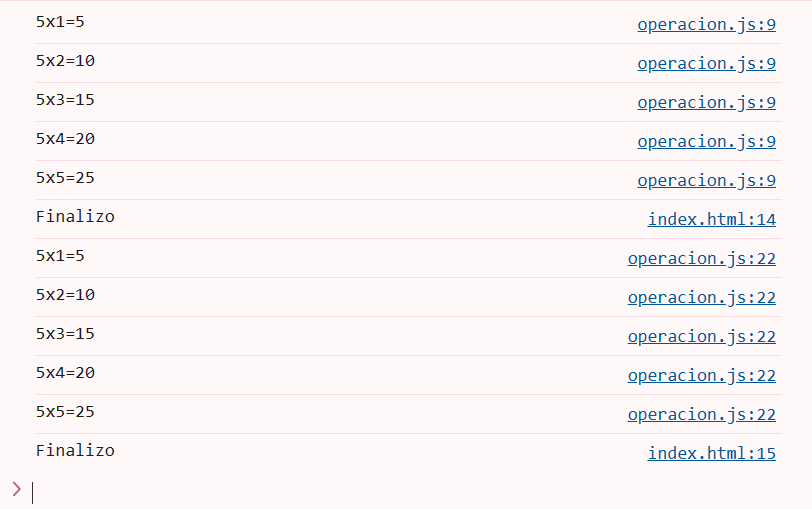
3.

While - mientras:

| **Nombre de la función: tablaMult(pmultiplicar,pnumero)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Realizar la tabla del 5 que multiplique hasta el 5 y debe imprimir los siguientes resultados:  5 x 1 = 5  5 x 2 = 10  5 x 3 = 15  5 x 4 = 20  5 x 5 = 25 | | |
| multiplicar  numero  result | Tipo de variable:  multiplicar(int)  numero (int)  result(int) | |
| Código:  function tablaMult(pmultiplicar,pnumero){      let multiplicar = pmultiplicar      let numero = pnumero      let result      while(multiplicar<numero){          multiplicar = multiplicar + 1          result = multiplicar \* numero          console.log(numero +"x"+ multiplicar +"="+ result);      }      return "Finalizo"  } | | |

| **Nombre de la función: tablaMultexp(pmultiplicar,pnumero)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Realizar la tabla del 5 que multiplique hasta el 5 y debe imprimir los siguientes resultados:  5 x 1 = 5  5 x 2 = 10  5 x 3 = 15  5 x 4 = 20  5 x 5 = 25 | | |
| multiplicar  numero  result | Tipo de variable:  multiplicar(int)  numero (int)  result(int) | |
| Código:  const tablaMultexp = function(pmultiplicar,pnumero){      let multiplicar = pmultiplicar      let numero = pnumero      let result      while(multiplicar<numero){          multiplicar = multiplicar + 1          result = multiplicar \* numero          console.log(numero +"x"+ multiplicar +"="+ result);      }      return "Finalizo"} | | |

Prueba de ambas:

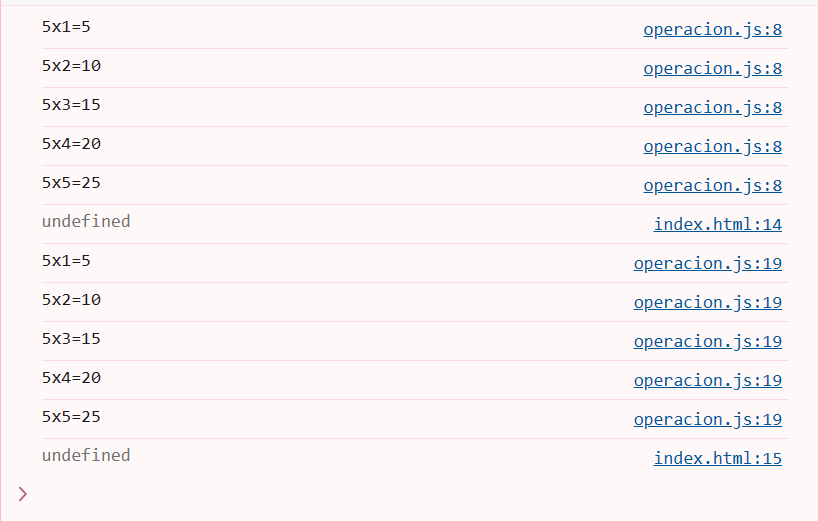


For - para:

| **Nombre de la función: tablaMult(pmultiplicar,pnumero)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Realizar la tabla del 5 que multiplique hasta el 5 y debe imprimir los siguientes resultados:  5 x 1 = 5  5 x 2 = 10  5 x 3 = 15  5 x 4 = 20  5 x 5 = 25 | | |
| multiplicar  numero  result | Tipo de variable:  multiplicar(int)  numero (int)  result(int) | |
| Código:  function tablaMult(pmultiplicar,pnumero){      let multiplicar = pmultiplicar      let numero = pnumero      let result      for(multiplicar = 1; multiplicar<=numero; multiplicar++){          result = multiplicar \* numero          console.log(numero +"x"+ multiplicar +"="+ result)      }      return "Finalizo"  } | | |

| **Nombre de la función: tablaMultexp(pmultiplicar,pnumero)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Realizar la tabla del 5 que multiplique hasta el 5 y debe imprimir los siguientes resultados:  5 x 1 = 5  5 x 2 = 10  5 x 3 = 15  5 x 4 = 20  5 x 5 = 25 | | |
| multiplicar  numero  resultado | Tipo de variable:  multiplicar(int)  numero (int)  resultado(int) | |
| Código:  const tablaMultexp = function(pmultiplicar,pnumero){      let multiplicar = pmultiplicar      let numero = pnumero      let result      for(multiplicar = 1; multiplicar<=numero; multiplicar++){          result = multiplicar \* numero          console.log(numero +"x"+ multiplicar +"="+ result)      }      return "Finalizo"  } | | |

Prueba de ambas:



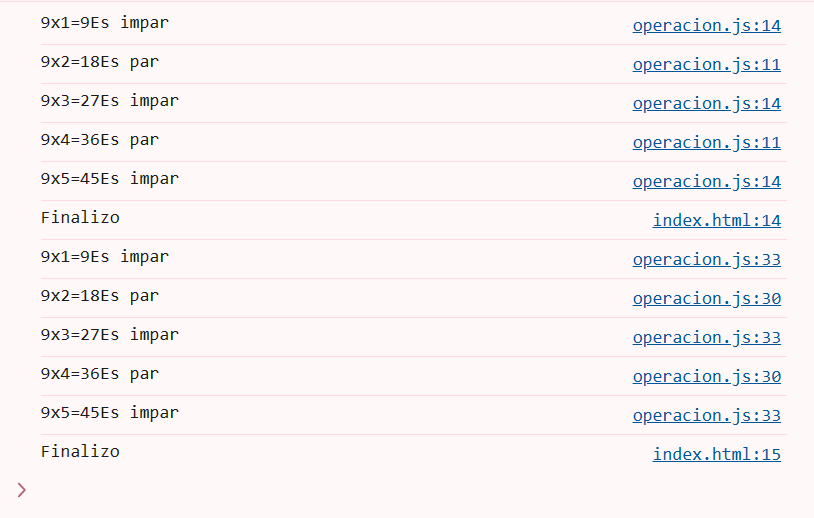
4.

While - mientras:

| **Nombre de la función: tablaMult(pnumero1,pnumero2)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Realizar la tabla del 5 que multiplique hasta 5 y den los resultados que son pares e impares. | | |
| Mult  Numero1  resultado  numero2 | Tipo de variable:  mult(int)  numero1 (int)  numero2(int)  resultado | |
| function tablaMult(pnumero1, pnumero2){         let mult=0          let numero1=pnumero1          let numero2=pnumero2          while(mult<numero2){              mult++              resultado=mult\*numero1              if(resultado%2==0){                  console.log(numero1 + "x" +  mult + "=" + resultado +  "Es par")              }              else{                  console.log(numero1 + "x" +  mult + "=" + resultado +  "Es impar)              }          }          return("Finalizo")  } | | |

| **Nombre de la función: tablaMultexp(pnumero1,pnumero2)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Realizar la tabla del 5 que multiplique hasta 5 y den los resultados que son pares e impares. | | |
| Mult  Numero1  resultado  numero2 | Tipo de variable:  mult(int)  numero1 (int)  numero2(int)  resultado | |
| const tablaMultexp=function(pnumero1, pnumero2){      let mult=0       let numero1=pnumero1       let numero2=pnumero2       while(mult<numero2){           mult++           resultado=mult\*numero1           if(resultado%2==0){               console.log(numero1 + "x" +  mult + "=" + resultado +  "Es par")           }           else{               console.log(numero1 + "x" +  mult + "=" + resultado +  "Es impar")           }       }       return("Finalizo")  } | | |

Prueba de ambas:



For - para:

| **Nombre de la función: tablaMult(pnumero1,pnumero2)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Realizar la tabla del 5 que multiplique hasta 5 y den los resultados que son pares e impares. | | |
| mult  numero1  numero2  resultado | Tipo de variable:  mult(int)  numero1 (int)  numero2(int)  resultado | |
| Código:  function tablaMult(pnumero1, pnumero2){      let numero1=pnumero1      let numero2=pnumero2          for(mult=1; mult<=numero2; mult++){              result=mult\*numero1              if(result%2==0){                  console.log(numero1 + "x" + mult + "=" + result + "Es par")              }else{                  console.log(numero1 + "x" + mult + "=" + result + "Es impar")              }          }          return("Finalizo")  } | | |

| **Nombre de la función: tablaMultexp(pnumero1,pnumero2)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Realizar la tabla del 5 que multiplique hasta 5 y den los resultados que son pares e impares. | | |
| mult(int)  numero1 (  numero2(  resultado | Tipo de variable:  mult(int)  numero1 (int)  numero2(int)  resultado | |
| const tablaMultexp=function(pnumero1, pnumero2){      let numero1=pnumero1      let numero2=pnumero2          for(mult=1; mult<=numero2; mult++){              result=mult\*numero1              if(result%2==0){                  console.log(numero1 + "x" + mult + "=" + result + "Es par")              }else{                  console.log(numero1 + "x" + mult + "=" + result + "Es impar")              }          }          return("Finalizo")    } | | |

Prueba de ambas:

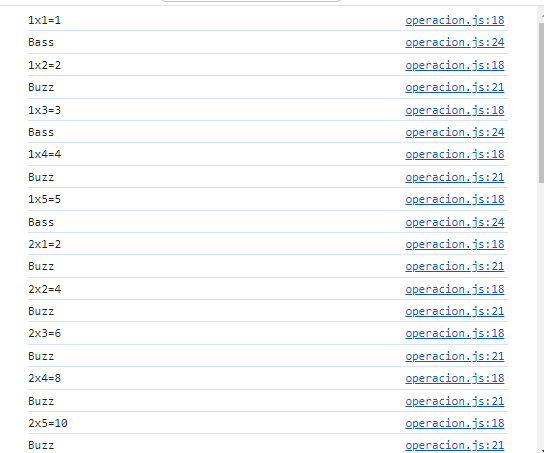


5.

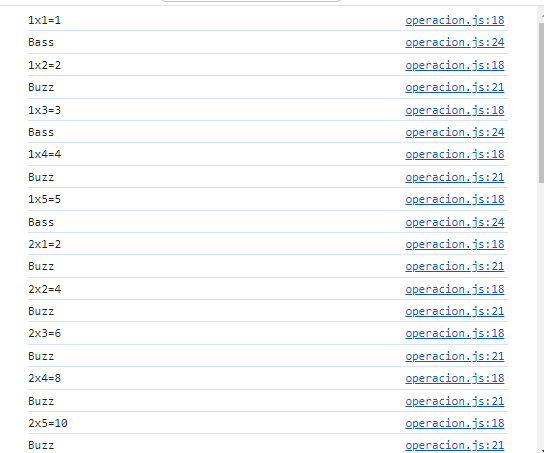
While - mientras:

| **Nombre de la función: tabla(ptabla1,ptabla2,plimit1,plimit2,ptablaPar,ptablaImpar)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Realizar tablas de multiplicar de 1 a 5 & multiplique hasta 5, debe imprimir los siguientes resultados a.  1X1=1 1x2=2 5X5=25 b. Cuando los resultados sean pares debe imprimir buzz Cuando sean impares debe imprimir bass. C. Debe imprimir Cuantos números sean pares e impares hay en los resultados de las multiplicaciones. | | |
| tabla1  tabla2  limit1  limit2  tablaPar  tablaImpar  result | Tipo de variable:  tabla1(int)  tabla2(int)  limit1(int)  limit2(int)  tablaPar(int)  tablaImpar(int)  result(int) | |
| function tabla(ptabla1,ptabla2,plimit1,plimit2,ptablaPar,ptablaImpar){      let tabla1 = ptabla1      let tabla2 = ptabla2      let limit1 = plimit1      let limit2 = plimit2      let tablaPar = ptablaPar      let tablaImpar = ptablaImpar      tabla1 = 0      while(tabla1<limit1){          tabla1 ++          tabla2 = 0          while(tabla2<limit2){              tabla2++              result = tabla1 \* tabla2              console.log(tabla1 +"x"+ tabla2 +"="+ result)              if(result%2==0){                  console.log("Buzz")                  tablaPar = tablaPar + 1              }              else{                  console.log("Bass")                  tablaImpar = tablaImpar + 1              }          }      }      return "Pares: " + tablaPar + " Impar: "+ tablaImpar +"\n"+ "Finalizo"  } | | |

| **Nombre de la función: tablaExp(ptabla1,ptabla2,plimit1,plimit2,ptablaPar,ptablaImpar)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Realizar tablas de multiplicar de 1 a 5 & multiplique hasta 5, debe imprimir los siguientes resultados a.  1X1=1 1x2=2 5X5=25  b. Cuando los resultados sean pares debe imprimir buzz Cuando sean impares debe imprimir bass. C. Debe imprimir Cuantos números sean pares e impares hay en los resultados de las multiplicaciones. | | |
| tabla1  tabla2  limit1  limit2  tablaPar  tablaImpar  result | Tipo de variable:  tabla1(int)  tabla2(int)  limit1(int)  limit2(int)  tablaPar(int)  tablaImpar(int)  result(int) | |
| const tablaExp = function(ptabla1,ptabla2,plimit1,plimit2,ptablaPar,ptablaImpar){      let tabla1 = ptabla1      let tabla2 = ptabla2      let limit1 = plimit1      let limit2 = plimit2      let tablaPar = ptablaPar      let tablaImpar = ptablaImpar      tabla1 = 0      while(tabla1<limit1){          tabla1 ++          tabla2 = 0          while(tabla2<limit2){              tabla2++              result = tabla1 \* tabla2              console.log(tabla1 +"x"+ tabla2 +"="+ result)              if(result%2==0){                  console.log("Buzz")                  tablaPar = tablaPar + 1              }              else{                  console.log("Bass")                  tablaImpar = tablaImpar + 1              }          }      }      return "Pares: " + tablaPar + " Impar: "+ tablaImpar +"\n"+ "Finalizo"  } | | |

Prueba de ambos: 







For - para:

| **Nombre de la función: tabla(ptabla1,ptabla2,plimit1,plimit2,ptablaPar,ptablaImpar)** | | **Versión: 1.0** |
| --- | --- | --- |
| **Descripción:**  Realizar tablas de multiplicar de 1 a 5 & multiplique hasta 5, debe imprimir los siguientes resultados a.  1X1=1 1x2=2 5X5=25  b. Cuando los resultados sean pares debe imprimir buzz Cuando sean impares debe imprimir bass. C. Debe imprimir Cuantos números sean pares e impares hay en los resultados de las multiplicaciones. | | |
| tabla1  tabla2  limit1  limit2  tablaPar  tablaImpar  result | Tipo de variable:  tabla1(int)  tabla2(int)  limit1(int)  limit2(int)  tablaPar(int)  tablaImpar(int)  result(int) | |
| *function* *tabla*(ptabla1, ptabla2, plimit1, plimit2, ptablaPar, ptablaImpar) {  *let* tablaPar = ptablaPar;  *let* tablaImpar = ptablaImpar;  *for* (*let* tabla1 = ptabla1; tabla1 < plimit1; tabla1++) {  *for* (*let* tabla2 = ptabla2; tabla2 < plimit2; tabla2++) {  *let* resultado = tabla1 \* tabla2;              console.*log*(tabla1 + "x" + tabla2 + "=" + resultado);  *if* (resultado % 2 === 0) {                  console.*log*("Buzz");                  tablaPar++;              } *else* {                  console.*log*("Bass");                  tablaImpar++;              }          }      }  *return* "Pares: " + tablaPar + " Impar: " + tablaImpar + "\n" + "Finalizo";  }  *// Ejemplo de uso:*  console.*log*(*tabla*(1, 1, 6, 6, 0, 0)); | | |

| **Nombre de la función: tablaExp(ptabla1,ptabla2,plimit1,plimit2,ptablaPar,ptablaImpar)** | | **Versión: 2.0** |
| --- | --- | --- |
| **Descripción:**  Realizar tablas de multiplicar de 1 a 5 & multiplique hasta 5, debe imprimir los siguientes resultados a.  1X1=1 1x2=2 5X5=25  b. Cuando los resultados sean pares debe imprimir buzz Cuando sean impares debe imprimir bass. C. Debe imprimir Cuantos números sean pares e impares hay en los resultados de las multiplicaciones. | | |
| tabla1  tabla2  limit1  limit2  tablaPar  tablaImpar  result | Tipo de variable:  tabla1(int)  tabla2(int)  limit1(int)  limit2(int)  tablaPar(int)  tablaImpar(int)  result(int) | |
| Código:  const tablaExp=function (*contador1*) {  *function* *tabla*(ptabla1, ptabla2, plimit1, plimit2, ptablaPar, ptablaImpar) {  *let* tablaPar = ptablaPar;  *let* tablaImpar = ptablaImpar;  *for* (*let* tabla1 = ptabla1; tabla1 < plimit1; tabla1++) {  *for* (*let* tabla2 = ptabla2; tabla2 < plimit2; tabla2++) {  *let* resultado = tabla1 \* tabla2;              console.*log*(tabla1 + "x" + tabla2 + "=" + resultado);  *if* (resultado % 2 === 0) {                  console.*log*("Buzz");                  tablaPar++;              } *else* {                  console.*log*("Bass");                  tablaImpar++;              }          }      }  *return* "Pares: " + tablaPar + " Impar: " + tablaImpar + "\n" + "Finalizo";  }  *// Ejemplo de uso:*  console.*log*(*tabla*(1, 1, 6, 6, 0, 0)); | | |

Prueba de ambas:

