

# Taller de Programación Web

MSc. Yuliana Jáuregui Rosas



Universidad  
Tecnológica  
del Perú

# Saberes previo

- ¿Como incluye un código CSS en una pagina web?
- ¿Para que se usa FlexBox?

# JavaScript



# Utilidad

- ¿Es importante que el usuario interactúe con la pagina web?

**Plataforma Digital de Cambio de Divisas**  
Ahorra con el mejor tipo de cambio, con la seguridad que buscas

AComo Compra  
S/ 3.8070



AComo Vende  
S/ 3.8220

USD    Tú Compras  
\$            0.00



PEN    Tú Depositas  
S/            0.00

Realizar operación

**Calcula el tamaño de la muestra**

Tamaño de la población ⓘ  
100000

Nivel de confianza (%) ⓘ  
95 ▼

Margen de error (%) ⓘ  
5

Tamaño de la muestra

0

Desaprende lo que te limita

# Logro



Al finalizar esta sesión el alumno  
desarrolla aplicaciones sobre  
navegadores utilizando el lenguaje  
JavaScript

# Temario



- JavaScript
- Implementación de JavaScript
- Constantes
- Variables
- Operadores
- Conversiones de tipo
- Funciones
- Estructuras condicionales
- Estructuras repetitivas

# JavaScript

- JavaScript es un lenguaje de programación con soporte para programación orientada a objetos, que permite añadir características interactivas a un sitio web, por ejemplo: juegos, eventos que ocurren al presionar los botones de un formulario, efectos de estilo dinámicos, animaciones y más.
- Fue creado en 1995 por Brendan Eich, co-fundador del proyecto Mozilla, Mozilla Foundation y la Corporación Mozilla, bajo el nombre de LiveScript, siendo luego nombrado como JavaScript.

# JavaScript

- Se dice que JavaScript es un **lenguaje del lado del cliente**, pues los scripts son ejecutados en el navegador del usuario.
- El **punto y coma** es un carácter clave en muchos lenguajes de programación pero en JavaScript su uso no es obligatorio, sin embargo se recomienda usarlo ya que indica el momento en que termina de ejecutarse una sentencia.
- JavaScript distingue entre mayúsculas y minúsculas.



# Implementación de JavaScript



- JavaScript se puede incorporar al documento mediante tres técnicas diferentes:
  - Como gestores en línea.
  - Como contenido del elemento `<script>`
  - Como un archivo externo.

# Gestores en línea

- El código se puede insertar en un elemento por medio de atributos. Este caso no es una buena practica.

```
<html>
  <head>
    <title>TODO supply a title</title>
    <meta charset="UTF-8">
  </head>
  <body>
    <button onclick="alert('Bienvenidos')">Aceptar</button>
  </body>
</html>
```

# Elemento `<script>`

- Se utiliza para trabajar con códigos extensos y personalizar las funciones

```
<html>
  <head>
    <title>TODO supply a title</title>
    <script>
      alert('Bienvenidos');
    </script>
  </head>
  <body>
    <div>TODO write content</div>
  </body>
</html>
```

# Archivo Externo

- Se crea un archivo con extensión **.js** que tendrá el código JavaScript, el cual podrá ser usado por varios documentos.
- Por otro lado, mediante el atributo **src** de la etiqueta **<script>** se asigna la ruta del archivo que contiene el código JavaScript.

micodigo.js

```
alert("Bienvenidos");
```

```
<html>
  <head>
    <title>TODO supply a title</title>
    <meta charset="UTF-8">
    <script src="micodigo.js"></script>
  </head>
  <body>
    <div>TODO write content</div>
  </body>
</html>
```

# Comentarios

- A la hora de programar es conveniente añadir comentarios para poder saber que es lo que hace cada parte del código.

## Comentario de una sola línea

```
//Esto es un comentario de una línea
```

## Comentario de varias líneas

```
/* Esto es un comentario que tiene mas de  
una línea es decir, varias líneas */
```

# Variables

- Las variables en JavaScript se declaran con la palabra clave **var** o **let** seguida del nombre que queremos asignarle. Asimismo, se puede asignar un valor inicial a través del signo igual (=).
- La diferencia principalmente entre var y let, radica en que let previene la sobreescritura de variables.
- **Ejemplos:**  
    var miVariable;  
    let miDato = 7;  
    let edad;
- En JavaScript, las variables son **tipadas** dinámicamente, esto significa que puedes asignarle primero un texto y luego un número cualquiera y sin restricciones.

# Variables

## Numéricas

- **Representa cualquier número entero, real, positivo o negativo.**  
`var miNumero = 2;`

## Cadenas

- **Representa un texto, se puede asignar un valor usando comillas dobles o comillas simples (apóstrofes).**  
`var miTexto1 = "Bienvenidos";`  
`var miTexto2 = 'Aprendiendo JavaScript';`

## Booleanos

- **Permite dos estados, verdadero o falso**  
`var valor1 = true;`  
`var valor2 = false;`

# Operadores aritméticos

Símbolo	Definición	Expresión	Resultado $a = 5$ y $b = 2$
+	Adición	$r = a + b;$	7
-	Sustracción	$r = a - b;$	3
*	Multiplicación	$r = a * b;$	10
/	División	$r = a / b;$	2
%	Resto	$r = a \% b;$	1
++	Incremento	$a++$	6
--	Decremento	$b--$	1



# Operadores relacionales

Símbolo	Definición	Expresión	Resultado $a = 10$ y $b = 7$
$>$	Mayor que	$a > b$	Verdadero
$>=$	Mayor o igual que	$a >= b$	Verdadero
$<$	Menor que	$a < b$	Falso
$<=$	Menor o igual que	$a <= b$	Falso
$==$	Igual a	$a == b$	Falso
$===$	Valor y tipo igual	$a === b$	Falso
$!=$	Diferente a	$a != b$	Verdadero
$!==$	Valor o tipo no igual	$a !== b$	Verdadero

# Operadores lógicos

Símbolo	Definición
&&	and
	or
!	not

x	y	x && y	x    y	!x
Verdadero	Verdadero	Verdadero	Verdadero	Falso
Verdadero	Falso	Falso	Verdadero	Falso
Falso	Verdadero	Falso	Verdadero	Verdadero
Falso	Falso	Falso	Falso	Verdadero

# Operadores de asignación

Símbolo	Definición	Expresión	Resultado $x = 12$
=	Asigna a una variable, el resultado de una expresión o un valor que se encuentra al lado derecho del signo =	$A = x + 5$	$A = 17$

# Operadores de asignación

- Se puede usar el operador de asignación compuesto cuando la operación contiene como primer operando la misma variable en la que se almacena el resultado.

Símbolo	Expresión	Expresión abreviada	Resultado a = 10
+=	a=a+6	a+=6	16
-=	a=a-5	a-=5	5
*=	a=a*4	a*=4	40
/=	a=a/2	a/=2	5
%=	a=a%3	a%=3	1

# Conversiones de tipo

**De texto a número entero:**

`nroEntero = parseInt(texto)`

**De texto a real:**

`nroReal = parseFloat(texto)`

# Eventos

- Los eventos son acciones u ocurrencias que suceden en el sistema que está programando y que el sistema le informa para que pueda responder de alguna manera.
- Por ejemplo, si el usuario hace clic en un botón en una página web, es posible que desee responder a esa acción mostrando un cuadro de información.

# Eventos

Evento	Descripción	Elementos para los que está definido
onblur	Deseleccionar el elemento	<button>, <input>, <label>, <select>, <textarea>, <body>
onchange	Deseleccionar un elemento que se ha modificado	<input>, <select>, <textarea>
onclick	Pinchar y soltar el ratón	Todos los elementos
ondblclick	Pinchar dos veces seguidas con el ratón	Todos los elementos
onfocus	Seleccionar un elemento	<button>, <input>, <label>, <select>, <textarea>, <body>
onkeydown	Pulsar una tecla (sin soltar)	Elementos de formulario y <body>
onkeypress	Pulsar una tecla	Elementos de formulario y <body>
onkeyup	Soltar una tecla pulsada	Elementos de formulario y <body>

# Eventos

Evento	Descripción	Elementos para los que está definido
onload	La página se ha cargado completamente	<body>
onmousedown	Pulsar (sin soltar) un botón del ratón	Todos los elementos
onmousemove	Mover el ratón	Todos los elementos
onmouseout	El ratón "sale" del elemento (pasa por encima de otro elemento)	Todos los elementos
onmouseover	El ratón "entra" en el elemento (pasa por encima del elemento)	Todos los elementos
onmouseup	Soltar el botón que estaba pulsado en el ratón	Todos los elementos



# Eventos

Evento	Descripción	Elementos para los que está definido
onreset	Inicializar el formulario (borrar todos sus datos)	<form>
onresize	Se ha modificado el tamaño de la ventana del navegador	<body>
onselect	Seleccionar un texto	<input>, <textarea>
onsubmit	Enviar el formulario	<form>
onunload	Se abandona la página (por ejemplo al cerrar el navegador)	<body>

# Funciones

- Una función es un pedazo de código que permanece inactivo hasta que es llamada a través de su nombre.
- Las funciones se declaran usando la palabra clave **function**, el nombre seguido de paréntesis, y el código entre llaves.
- Ejemplo:

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
  <script>
    function mostrarMensaje() {
      alert("Soy una función");
    }
  </script>
</head>
<body>
  <button onclick="mostrarMensaje()">Mostrar</button>
</body>
```

# Funciones

- Cuando las funciones también se llaman mediante un enlace, se debe incluir la palabra **javascript** seguido del **nombre de la función** en la propiedad **href**.

```
<body>
  <a href="javascript:msj()">Mensaje</a>
  <a href="javascript:window.print()">Imprimir</a>
  <a href="javascript:window.location.reload()">Refrescar</a>
</body>
```

# Ámbito de las variables

- Las variables declaradas fuera de una función se considera que están en un **ámbito global**, por lo tanto, pueden ser usadas desde cualquier parte del código.
- Las variables declaradas dentro de las funciones tienen un **ámbito local**, lo que significa que estas variables solo pueden ser usadas dentro de la función en la que fueron declaradas.

```
let variableGlobal = 5;
function mifuncion1(){
    let variableLocal = "El valor es ";
    alert(variableLocal + variableGlobal);
}
mifuncion1();
function mifuncion2(){
    let variableLocal = 7;
    alert("Nuevo valor: "+(variableLocal-variableGlobal));
}
mifuncion2();
```

# Funciones propias de JavaScript

- `alert()`
- `prompt()`
- `document.write()`
- `confirm()`
- `document.getElementById()`
- `document.getElementsByName()`

# Función alert()

- Muestra el valor del parámetro a través de una ventana emergente con un botón.
- Su sintaxis es:

```
alert("Texto a mostrar");
```

Donde:

- **Texto a mostrar:** Es el mensaje que se mostrará en la ventana emergente.

# Ejemplo 1

- Mostrar un mensaje de bienvenida.

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
  <script type="text/javascript" src="js/miscript.js">
  </script>
</head>
<body>
  <button onclick="ejemplo1()">Saludo</button><br>
</body>
</html>
```

```
function ejemplo1 () {
  alert("Bienvenidos a la clase de JavaScript")
}
```

# Función prompt()

- Esta función muestra una ventana emergente con un campo de entrada que permite al usuario introducir un valor. Retorna el valor ingresado. Su sintaxis es:

```
prompt("Texto descriptivo","Texto por defecto");
```

Donde:

- **Texto descriptivo:** Es el mensaje que se mostrará por pantalla pidiendo la inserción o rellanado de información.
- **Texto por defecto (Opcional):** Podemos dejar un valor por defecto para que el usuario no tenga que escribirlo o rellenarlo.



# Ejemplo 2

- Ingresar el nombre de una persona y mostrar un saludo con el nombre ingresado

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
  <script type="text/javascript" src="js/miscript.js">
  </script>
</head>
<body>
  <button onclick="ejemplo1()">Saludo</button><br>
  <button onclick="ejemplo2()">Saludo personalizado</button><br>
</body>
</html>
```

```
function ejemplo2 () {
  let nombre;
  nombre=prompt("Ingrese Nombre:", "");
  alert("Bienvenid@ "+nombre);
}
```

localhost:8383 dice

Ingrese Nombre:

Aceptar Cancelar

localhost:8383 dice

Bienvenid@: Yuliana

Aceptar

Desaprende lo que te limita

# Función document.write()

- La función document.write() escribe el contenido solo al cargar la página, si se llama posteriormente, será necesario volverla a cargar.
- Su sintaxis es:

```
document.write('texto');  
document.write(variable);  
document.write(variable1 + variable2);
```

Donde:

- **Texto:** es el mensaje que se muestra.
- **Variable:** muestra el valor que contiene una variables

# Ejemplo 3

- Calcular la suma de 2 números enteros.

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
  <script type="text/javascript" src="js/miscrypt.js">
  </script>
</head>
<body>
  <button onclick="ejemplo1()">Saludo</button><br>
  <button onclick="ejemplo2()">Saludo personalizado</button><br>
  <button onclick="ejemplo3()">Suma de numeros</button><br>
</body>
</html>
```

```
function ejemplo3 () {
  let n1, n2, suma;
  n1=prompt("Ingresar 1er número");
  n2=prompt("Ingresar 2do número");
  suma=parseInt(n1)+parseInt(n2);
  document.write("La suma es "+suma);
}
```

# Ejemplo 4

- Mostrar una imagen a través de un botón

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
  <script type="text/javascript" src="js/miscript.js">
  </script>
</head>
<body>
  <button onclick="ejemplo1()">Saludo</button><br>
  <button onclick="ejemplo2()">Saludo personalizado</button><br>
  <button onclick="ejemplo3()">Suma de numeros</button><br>
  <button onclick="ejemplo4()">Mostrar imagen</button><br>
</body>
</html>
```

```
function ejemplo4 () {
  document.write('');
}
```

# Función confirm()

- Muestra una ventana de diálogo con un mensaje opcional y dos botones, Aceptar y Cancelar.
- Su sintaxis es:

```
resultado = confirm('mensaje');
```

Donde:

- **Mensaje:** es la cadena que se muestra en el diálogo.
- **Resultado:** es un valor booleano indicando si se ha pulsado Aceptar (true) o Cancelar (false).

# Ejemplo 5

- Confirmar si desea visitar la pagina de la UTP.

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
  <script type="text/javascript" src="js/miscrypt.js">
  </script>
</head>
<body>
  <button onclick="ejemplo1()">Saludo</button><br>
  <button onclick="ejemplo2()">Saludo personalizado</button><br>
  <button onclick="ejemplo3()">Suma de numeros</button><br>
  <button onclick="ejemplo4()">Mostrar imagen</button><br>
  <button onclick="ejemplo5()">Visitar pagina UTP</button><br>
</body>
</html>
```

```
function ejemplo5 () {
  let respuesta = confirm("¿Desea visitar la pagina de UTP?")
  if(respuesta){
    alert("Bien!!, buena elección")
    window.location = "https://www.utp.edu.pe/";
  }
  else{
    alert("OK, será en otro momento");
  }
}
```

Desaprende lo que te limita

# Función `document.getElementById()`

- Esta función pertenece al objeto **document**. Permite obtener el objeto que hace referencia al elemento con un id concreto.
- Permite obtener un valor numérico si se considera el siguiente formato.

```
variable = document.getElementById(id).value;
```

# Función `document.getElementsByName()`

- Permite obtener un conjunto de valores (array).

```
arreglo = document.getElementsByName(name);
```



# Ejemplo 6

- Ingresar el nombre, edad, genero, estado civil, hobbies de una persona y mostrar estos datos ingresados en una área de texto.

Nombre:	<input type="text"/>
Edad:	<input type="text"/>
Genero:	<input checked="" type="radio"/> Masculino <input type="radio"/> Femenino
Estado civil:	<input type="text" value="Soltero"/> ▼
Hobbies:	<input type="checkbox"/> Musica <input type="checkbox"/> Lectura <input type="checkbox"/> Canto
Respuestas	<input type="text"/>
<input type="button" value="Mostrar"/>	

# Ejemplo



Universidad  
Tecnológica  
del Perú

```
<form name="Form2">
  <label>Ingrese nombre:</label>
  <input type="text" id="nombre"><br>
  <label>Género:</label>
    <input type="radio" name="genero" value="Masculino" checked>Masculino
    <input type="radio" name="genero" value="Femenino">Femenino <br>
  <label>Grado de instrucción:</label>
    <select id="instruccion">
      <option>Primaria</option>
      <option>Secundaria</option>
      <option>Superior</option>
    </select><br>
  <label>Hobbies:</label><br>
    <input type="checkbox" name="musica">Música<br>
    <input type="checkbox" name="lectura">Lectura<br>
    <input type="checkbox" name="pelicula">Películas<br>
    <input type="checkbox" name="deporte">Deporte<br>
  <input type="button" value="Mostrar" onclick="ejem2()"><br><br>
  <label>Datos ingresados:</label><br>
  <textarea name="respuesta" rows="6" cols="35"></textarea><br>
</form>
```

Desaprende lo que te limita

# Ejemplo



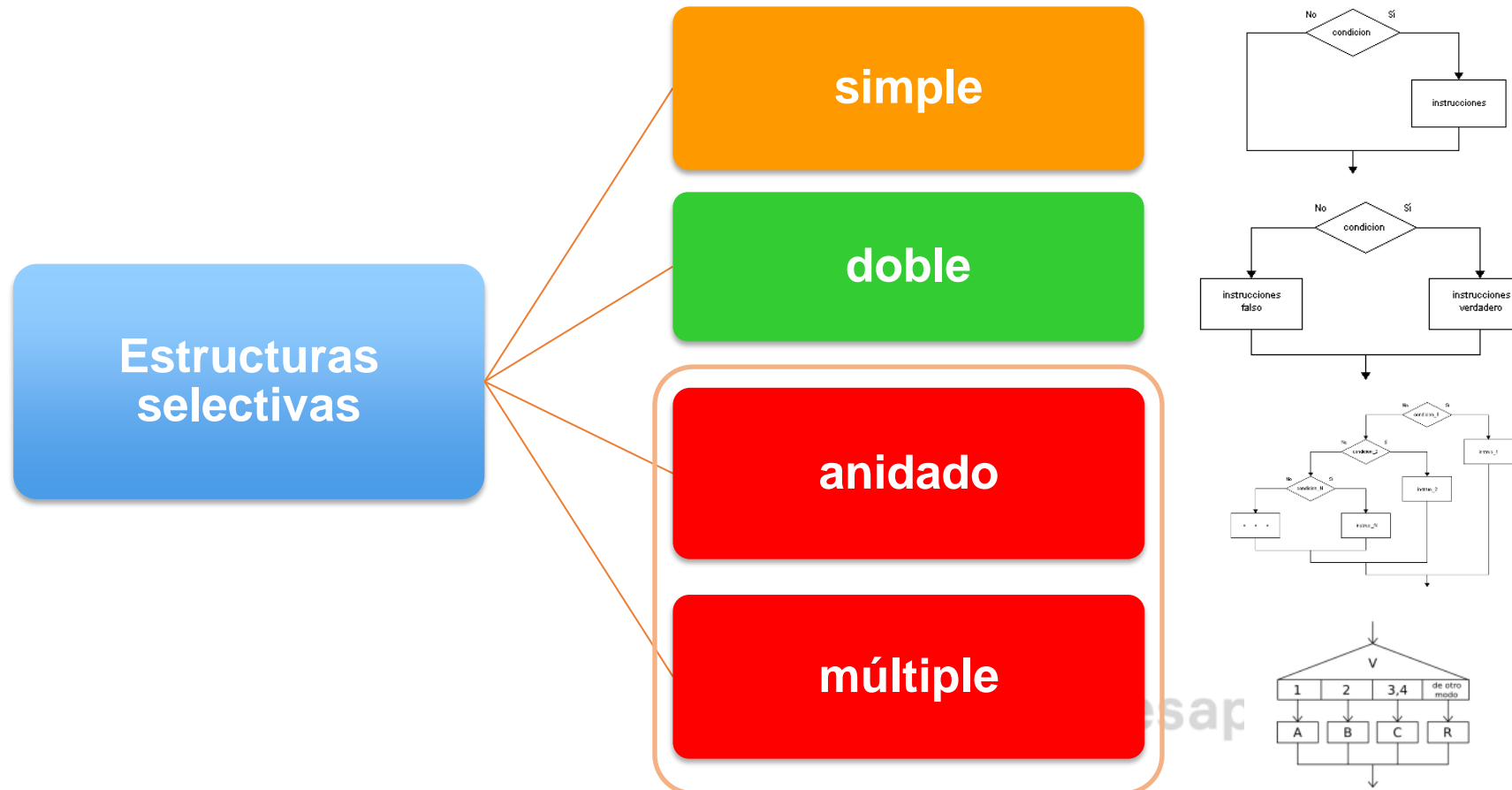
Universidad  
Tecnológica  
del Perú

```
function ejem2(){  
    var nom=document.getElementById('nombre').value;  
    var gen, hobbie1="",hobbie2="", hobbie3="", hobbie4="";  
    var porNombre=document.getElementsByName("genero");  
    for(var i=0; i<porNombre.length; i++)  
        if(porNombre[i].checked)  
            gen=porNombre[i].value;  
    var inst=document.getElementById('instruccion').value;  
    if(document.getElementsByName("musica")[0].checked===true)  
        var hobbie1="Música";  
    if(document.getElementsByName("lectura")[0].checked===true)  
        var hobbie2="Lectura";  
    if(document.getElementsByName("pelicula")[0].checked===true)  
        var hobbie3="Películas";  
    if(document.getElementsByName("deporte")[0].checked===true)  
        var hobbie4="Deporte";  
    document.Form2.respuesta.value = "Su nombre es: "+nom+  
        "\nSu genero es: "+gen+  
        "\nSu grado de instruccion es: "+inst+  
        "\nSus hobbies son: \n"+  
        hobbie1+" "+hobbie2+" "+hobbie3+" "+hobbie4;  
}
```

o que te limita

# Estructuras condicionales

- Una estructura condicional permite elegir entre una, dos o más alternativas diferentes, el cual dependerá de evaluar una condición cuya respuesta puede ser **Verdadera** o **Falsa**.



# Estructura Condicional Simple - if



Universidad  
Tecnológica  
del Perú

- La estructura condicional simple evalúa una condición (expresión lógica) y si es verdadera ejecuta la(s) instrucción(es) correspondientes.
- De ser falsa la condición, simplemente no hace nada.
- Su sintaxis es:

```
if (condición) {  
    instrucción;  
}
```

Desaprende lo que te limita

# Ejercicio 1

- Un parque de diversiones ofrece descuentos del 15% a sus clientes cuya edad sea menor a 12 años. Mostrar el pago final de un cliente ingresando su edad.

# Estructura Condicional Doble (if – else)

Permite que el flujo de control de un programa elija entre dos posibles bloques de instrucciones, esto es:

- Si la condición (expresión lógica) es verdadera, entonces ejecuta el bloque de instrucciones que viene a continuación.
- Si la condición es falsa, ejecuta otro bloque de instrucciones.
- Su sintaxis es:

```
if (condición) {  
    instrucción1;  
}  
else {  
    instrucción2;  
}
```

## Ejercicio 2

- Elaborar un programa que calcule el promedio final de un alumnos en base a 3 notas y diga si el alumnos aprobó o desaprobó, considere que el alumno se encuentra aprobado si obtiene un promedio superior o igual a 12.



# Estructura Condicional anidada

- Un if anidado es una sentencia if que esta contenido dentro de otro if o dentro de un else. Cuando se anidan if, cada sentencia else siempre se corresponde a la sentencia if mas próxima que no esté asociada con otro else.
- Su sintaxis es:

```
if (condición1) {  
    instruccion1;  
}  
else  
    if (condición2) {  
        instruccion2;  
    }  
    else  
        if (condición3) {  
            instruccion3;  
        }  
        else {  
            instruccion4;  
        }  
}
```

# Ejercicio 3

- Elabore un programa que determine la calificación de un estudiante en base a su promedio, de acuerdo a la siguiente tabla:

Promedio	Calificación
20 – 18	A
17 – 14	B
13 – 11	C
10 – 6	D
5 – 0	E

# Estructura Condicional múltiple

- Esta estructura ejecuta una acción dependiendo del resultado del selector. Se presenta como una alternativa de la sentencia Si anidada, ya que en ocasiones resulta ser más comprensible y ordenado.
- Su sintaxis es:

```
switch (variable) {  
    case c1: instruccion1;    break;  
    case c2: instruccion2;    break;  
    ...  
    case cn: instruccionN;    break;  
    default: instruccionX;  
}
```

# Ejercicio 4

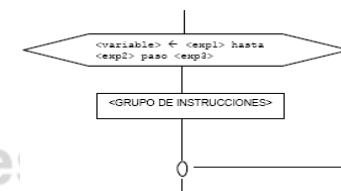
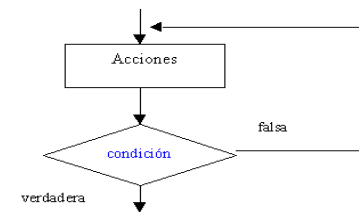
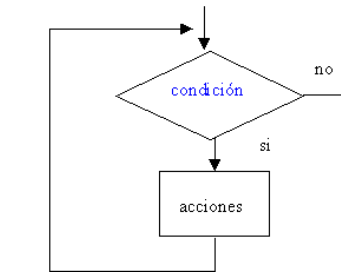
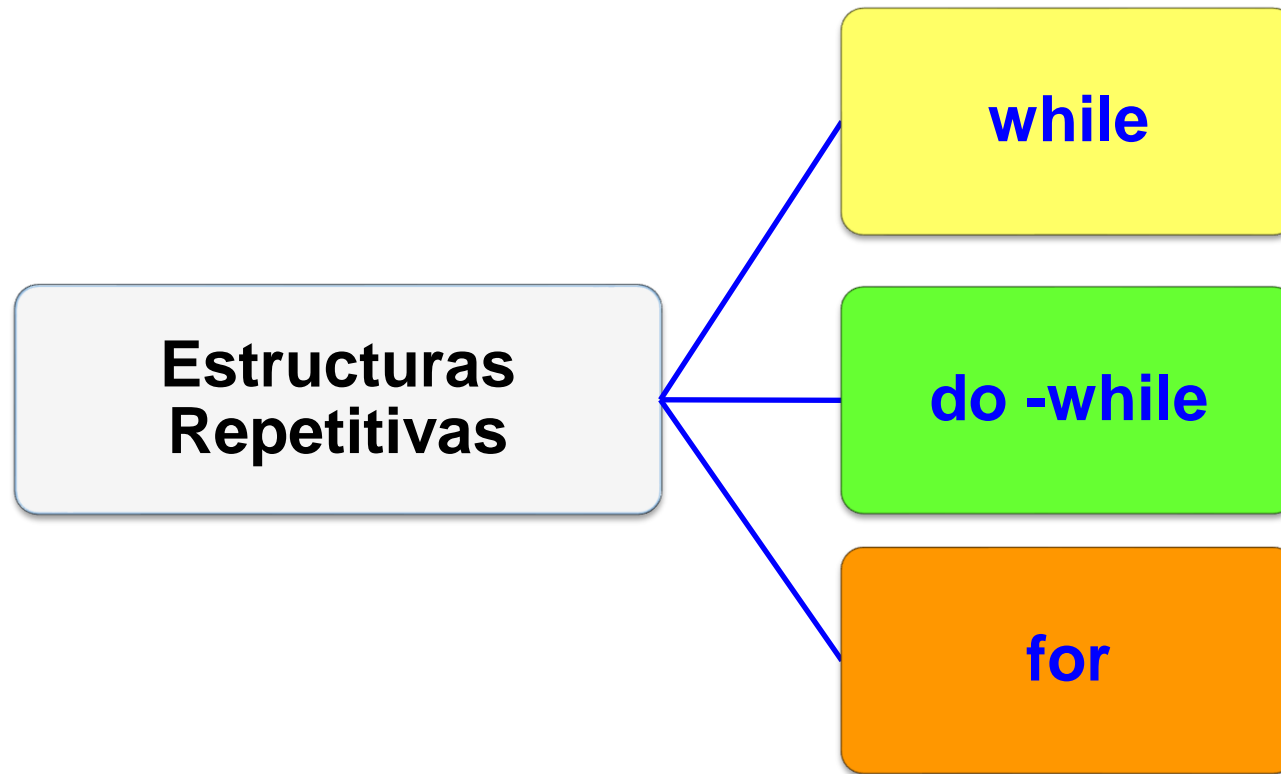
- Escriba un programa que permita ingresar un número entero entre 1 y 7 y muestre el día correspondiente, considerando que 1 representa al Lunes y 7 a Domingo.

# Ejercicio 5

- Escriba un programa que al ingresar un mes devuelva la cantidad de días que trae el mes.

# Estructuras repetitivas

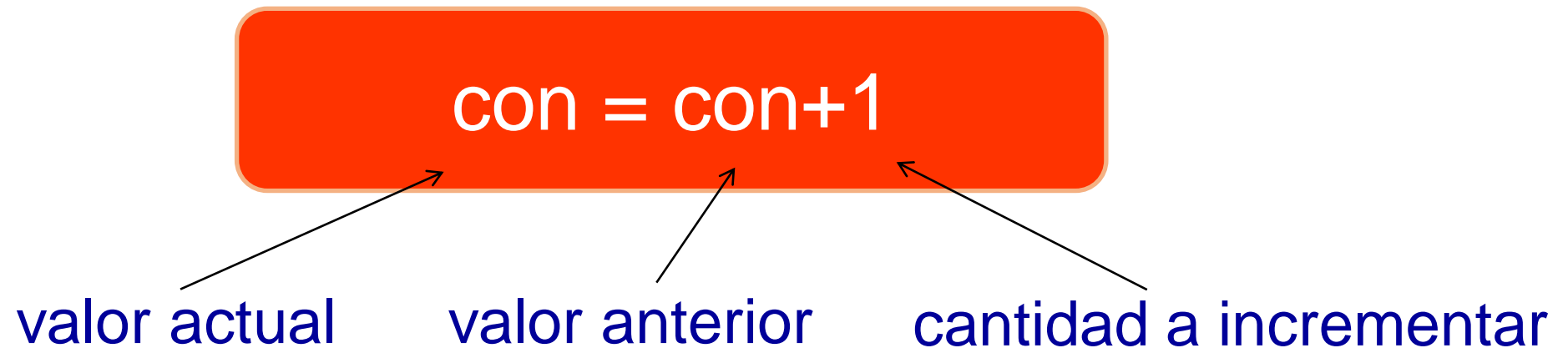
- Las estructuras que repiten una secuencia de instrucciones un número determinado de veces se denominan **Bucles**. Entre las estructuras repetitivas se encuentran:



De... que te limita

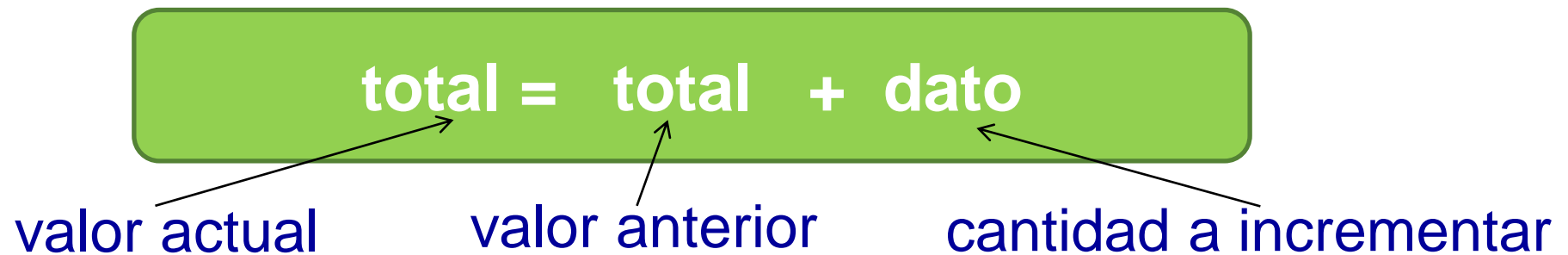
# Contador

- Es una variable numérica entera que se incrementa o decrementa cada vez que se ejecuta la acción que lo contiene, toma un valor inicial de cero o uno, según sea el caso.



# Acumulador

- Es una variable numérica que se incrementa o decrementa de forma no constante, toma un valor inicial de cero o uno según sea el caso. En la siguiente expresión, la variable total representa a un acumulador y la variable dato es la cantidad que se incrementará el acumulador en cada repetición.





# Estructura while

- La estructura repetitiva while, es aquella en que el cuerpo del bucle se repite mientras se cumple una determinada condición.
- Su sintaxis es:

```
while (condición){  
    instrucciones;  
}
```

# Ejemplo 6

- Una persona tomó el tiempo en minutos que demoró en llegar de su casa a su trabajo durante 6 días. Escriba un programa que permita ingresar cada uno de los tiempos y muestre el promedio de tiempos y el número de veces que se demoró menos de 35 minutos.

# Estructura do - while

- La instrucción do - while ejecuta una secuencia de instrucciones, repitiéndolas siempre que la condición (expresión lógica) sea verdadera.
- La ejecución finaliza cuando la condición (expresión lógica) es falsa.
- Mínimamente se ejecuta una vez las instrucciones que se encuentran dentro de la estructura do - while.
- Su sintaxis es:

```
do {  
    instrucciones;  
}  
while (condición);
```

# Ejemplo 7

- Elaborar un programa que permita ingresar una secuencia de números enteros y cuente la cantidad de números positivos y negativos que se ingresaron. El programa termina cuando se ingresa el numero cero.

# Estructura for

- La instrucción for ejecuta una secuencia de instrucciones, un número determinado de veces.

```
for (inicialización; condición; incremento){  
    instrucciones;  
}
```

# Ejemplo 8

- Elaborar un programa que calcule el promedio de los  $n$  primeros números enteros positivos.

# Ejemplo 9

- En un centro de salud pediátrica se registra el peso, la talla y el genero de cada uno de los  $N$  pacientes, se desea saber:
  - La cantidad de niños que pesan más de 17 kilos.
  - La Cantidad de niñas miden menos de 0.50 centímetros.
  - El peso promedio de las niñas.
  - La talla promedio de los niños.

# Ingreso y salida de datos desde formularios

- Para leer y escribir datos a través de los formularios es necesario trabajar con el atributo **name** tanto del formulario como del elemento.
- De esta manera para leer un dato se usa la siguiente sintaxis:

```
variable = document.nombreFormulario.nombreElemento.value ;
```

- Y para escribir un dato se usa la siguiente sintaxis:

```
document.nombreFormulario.nombreElemento.value = variable
```



# Ejemplo

- Calcular el producto de 2 números

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
  <script type="text/javascript" src="js/micodigo.js"></script>
</head>
<body>
  <form name="Form1">
    <label>Ingrese numero1: </label>
    <input type="text" id="n1" name="nro1"><br>
    <label>Ingrese numero2: </label>
    <input type="text" id="n2" name="nro2"><br>
    <label>Producto: </label>
    <input type="text" name="resultado"><br>
    <input type="button" value="Calcular" onclick="ejem1()">
  </form>
</body>
</html>
```

```
function ejem1()
{
  var x, y, z;
  x=document.Form1.nro1.value;
  y=document.Form1.nro2.value;
  z=x*y;
  document.Form1.resultado.value = z;
}
```

Ingrese numero1:

Ingrese numero2:

Producto:

# innerHTML

- La propiedad innerHTML permiten escribir elementos de forma dinámica, sin tener que recargar, por lo que más empleados actualmente.
- Para usar innerHTML es necesario identificar en primer lugar el elemento donde se va a escribir. La forma más fácil de hacerlo es asignar a un elemento un identificador (id). Después de eso se debe usar document.getElementById() y escribir el contenido.

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
  <script>
    function escribir(){
      document.getElementById('contenido').innerHTML='Buenas noches';
    }
  </script>
</head>
<body>
  <div id="contenido"></div>
  <a href="javascript:escribir()">Saludo</a>
</body>
</html>
```

# innerHTML



Universidad  
Tecnológica  
del Perú

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
  <script>
    function escribir2() {
      valor = document.getElementById('dato').value;
      document.getElementById('datoMostrado').innerHTML=' '+valor;
    }
  </script>
</head>
<body>
  <p>Ingresa una direccion: </p>
  <input id="dato" onkeyup="escribir2()">
  <div id="datoMostrado"></div>
</body>
</html>
```





Desaprende lo que te limita

# Template String

- Los template strings fueron introducidos en la especificación de ES2015 y nos permiten interpolar valores dentro de un string.

```
var nombre = 'Juan', trabajo = 'Desarrollador web';  
  
document.write('Nombre: '+nombre+', Trabajo: '+trabajo);  
document.write(`Nombre: ${nombre}, Trabajo: ${trabajo}`);
```

Salida:

		top			Filter
Nombre: Juan, Trabajo: Desarrollador web					
Nombre: Juan, Trabajo: Desarrollador web					

# Ejemplo

- Mostrar el nombre y las horas de trabajadas de un empleado.

```
<form name="miFormulario">
  <label>Ingrese nombre: </label>
  <input type="text" id="n1" name="nombre"><br>
  <label>Ingrese horas trabajadas: </label>
  <input type="text" id="n2" name="horas"><br>
  <p id="respuesta"><p>
  <input type="button" value="Calcular" onclick="Ejemplo()">
</form>
```

```
function Ejemplo(){
  var nom, hor;
  nom=document.miFormulario.nombre.value;
  hor=document.miFormulario.horas.value;
  //imprimiendo en la misma pagina
  document.getElementById('respuesta').innerHTML = `El empleado ${nom} trabajo ${hor} horas`+"<br>";
  //haciendo lo mismo pero simplificado
  respuesta.innerHTML += `El empleado ${nom} trabajo ${hor} horas`;
}
```

# Abrir nuevas ventanas

- A partir de un enlace o un botón de una página web se puede abrir una nueva ventana.

```
function popUp(URL) {  
    window.open(URL, 'Nombre de la ventana', 'width=600,height=400,left=50,top=50,toolbar=yes');  
}
```

```
<a href="javascript:popUp('https://www.utp.edu.pe/')">Nueva ventana</a>
```

# Parámetros window.open()

- Los parámetros que se pueden usar en la función window.open() son:

Parámetro	Descripción
width	Ajusta el ancho de la ventana (en píxels).
height	Ajusta el alto de la ventana (en pixels).
top	Margen con respecto a la parte superior de la pantalla (en píxels).
left	Margen con respecto a la parte izquierda de la pantalla (en píxels).
scrollbars	Muestra las barras de scroll (yes/no o 1-0).
resizable	Indica si se permite redimensionar la ventana (yes/no o 1-0).
location	Muestra la barra de direcciones (yes/no o 1-0).
status	Muestra la barra de estado (yes/no o 1-0).
titlebar	Muestra la barra de título (yes/no o 1-0).
toolbar	Muestra la barra de herramientas (yes/no o 1-0).
menubar	Muestra la barra de menú (yes/no o 1-0).

# Cambiando el estilo

- Javascript ofrece la posibilidad, modificar el estilo (tamaño, estilo, tipo de fuente, color de fondo, etc.) original de una página y sin tener que volver a cargar la página del servidor.
- Para que el usuario pueda realizar estas modificaciones puede utilizarse cualquier evento, como un clic del ratón o cualquier otro método con que se pueda transmitir una instrucción.
- La modificación al estilo puede ser accedido de varias formas, desde el **documento** o desde el **elemento** individual al que va dirigido el estilo.



# Desde el documento

- Permite aplicar estilo de forma general al documento.
- Su sintaxis es:

```
document.body.style.propiedad = "valor"
```

# Ejemplo

- Cambiar el tamaño de fuente y el color de fondo de toda la pagina.

```
<h2>Cambio el estilo a todo el documento</h2> <br>  
<a href="javascript:cambios1()">Cambiar fuente y fondo</a><br>  
<a href="javascript:cambios2()">Quitar cambios</a>
```

```
function cambios1() {  
    document.body.style.fontSize = '30px';  
    document.body.style.background = '#E4EDA5';  
}  
function cambios2() {  
    document.body.style.fontSize = '16px';  
    document.body.style.background = '#FFFFFF';  
}
```

# Desde el elemento

- Permite aplicar estilo a un bloque o elemento definido por un identificador (id).
- Su sintaxis es:

```
document.getElementById("id").style.propiedad = "valor"
```

# Ejemplo

- Cambiar tipo y color de fuente de un párrafo.

```
<p id="texto">Lorem ipsum dolor sit amet, consectetur adipisicing elit,
tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse
cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non
proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
</p>
```

```
<h2>Cambio el estilo a un parrafo</h2> <br>
<a href="javascript:cambios3()">Cambiar tipo y color de fuente</a><br>
<a href="javascript:cambios4()">Quitar cambios</a><br>
```

```
function cambios3() {
    document.getElementById('texto').style.fontFamily = 'Comic Sans MS';
    document.getElementById('texto').style.color = '#EA0D22' ;
}
function cambios4() {
    document.getElementById('texto').style.fontFamily = 'Arial';
    document.getElementById('texto').style.color = 'black' ;
}
```

# Ejemplo

- Cambiar el tipo de fuente al pasar el mouse sobre un texto.

```
<p id="parrafo">Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed  
tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,  
quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo  
consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse  
cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non  
proident, sunt in culpa qui officia deserunt mollit anim id est laborum.  
</p>  
  
<a href="#parrafo" onmouseover="cambios5()" onmouseout="cambios6()">Resaltar</a>
```

```
function cambios5() {  
    document.getElementById('parrafo').style.background = '#DEF017';  
    document.getElementById('parrafo').style.fontSize = '24px';  
}  
function cambios6() {  
    document.getElementById('parrafo').style.background = 'FFFFFF';  
    document.getElementById('parrafo').style.fontSize = '16px';  
}
```

# Arreglos

- Es un conjunto elementos o variables que se encuentran en posiciones consecutivas de memoria.
- Para acceder a los elementos se debe indicar el nombre del arreglo y su posición.
- La posición de un arreglo comienza en 0.
- Un arreglo puede contener distintos tipos de datos (cadena, numéricos, booleanos e incluso otros arreglos).
- Ejemplo:

```
var colores = ["rojo", "verde", "azul"];  
let notas = [15, 18, 9, 12];
```

# Arreglos

Los arreglos también se pueden definir mediante el operador new

- Creación de un arreglo:

```
meses = new Array(12);
```

- Creación de un arreglo con un número indefinido:

```
alumnos = new Array();
```

- Creación de un arreglo mediante la instrucción new Array

```
varios = new Array("c001", "Luis", 23, true);
```

# Mostrando Arreglos

- La función `alert()` puede mostrar no solo valores independientes, sino arrays completos.
- Si queremos ver todos los valores incluidos en el array, solo tenemos que especificar el nombre del array.

```
var colores = ['rojo', 'verde', 'azul'];  
alert(colores);           // "rojo, verde, azul"
```



# Ejemplos

- Se muestra el segundo elemento del vector

```
var vector = ["rojo", 32, "Trabajando con JavaScript"];  
alert(vector[1]);
```

- Se muestra el primer numero de la segunda fila

```
var matriz = [[2, 45, 31], [5, 10], [81, 12]];  
alert(matriz[1][0]);
```

# Funciones de arreglos

```
var meses = new Array('enero', 'febrero', 'abril');
```

Función	Descripción	Ejemplo
push()	Agrega un ítem al final del arreglo	meses.push('marzo');
unshift()	Agrega un ítem al inicio del arreglo.	meses.unshift('diciembre');
pop()	Elimina el ítem final del arreglo.	meses.pop();
shift()	Elimina el ítem inicial del arreglo.	meses.shift();
splice(x, y)	Elimina y agrega ítems a partir del índice x	meses.splice(0, 2);
reverse()	Muestra el arreglo al revés	meses.reverse();
sort()	Muestra el arreglo ordenado alfabéticamente	meses.sort();

# Objetos

- Un objeto es una estructura de código que modela un objeto de la vida real. Un objeto es una colección de propiedades, y una propiedad es una asociación entre un nombre (o clave) y un valor. Un objeto se define de la siguiente forma:

```
objeto = {atributo1: 'valor', atributo2: 'valor'}
```

- Un valor se recupera:

```
objeto.atributo
```

# Objetos

- El valor de una propiedad puede ser una función, o un valor conocido.

```
var persona={
  nombre:'Juan',
  apellido:'Perez',
  edad: 22,
  trabaja: true,
  musica: ['rock', 'balada', 'alternativo'],
  hogar:{
    ciudad:'Lima',
    pais: 'Peru',
  }
}
document.write(persona.hogar.pais);
document.write(persona['hogar']['pais']);
document.write(persona.musica);
document.write(persona.musica[1]);
```

```
let alumno ={
  codigo : 1235,
  nombre : "Carlos",
  edad : 22,
  telefono : "999756321",
  direccion : "Av. Brasil 725"
};
```

# Objetos

- Por ejemplo:

```
let alumno = { nombre : 'Cesar', edad : 21 };
```

- Para recuperar el nombre, se usa: alumno.nombre
- Para recuperar la edad, se usa: alumno.edad

# Uso del for in

- Es una alternativa al bucle for, permite recorrer las propiedades de los objetos en Javascript.

```
let alumno = {  
  nombre : 'Cesar',  
  edad : 21,  
  direccion : 'Av. Arequipa 235'};  
  
for (dato in alumno) {  
  document.writeln(dato, " ", alumno[dato]);  
}
```

# Clase

- Las clases son "funciones especiales", como las expresiones de funciones y declaraciones de funciones, la sintaxis de una clase tiene dos componentes: expresiones de clases y declaraciones de clases.

```
class tarea{  
    constructor(nombre, urgencia){  
        this.nombre=nombre;  
        this.urgencia=urgencia;  
    }  
}  
  
var tarea1 = new tarea('Aprende JavaScript', 'Urgente');  
var tarea2 = new tarea('Aprende Java', 'Avanzado');  
var tarea3 = new tarea('Aprende JSF', 'Urgente');  
document.write(tarea1);  
document.write(tarea2);  
document.write(tarea3);
```

# Ejercicio

- Construir un programa que registre los galones de gasolina que vende un grifo, calcular el monto que paga cada cliente y el total que recauda el grifo, el precio de cada galón de gasolina varia de acuerdo al siguiente cuadro:

Tipo	Precio (S/.)
A	15.20
B	14.50
C	13.30

Cantidad de galones:

Pago

Tipo de gasolina:

☐ A

☐ B

☐ C

Total





**Universidad  
Tecnológica  
del Perú**