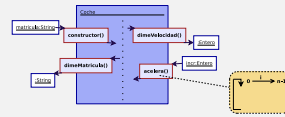


Grado Tecnologías Interactivas

Programación 2



Práctica 1

UNIVERSIDAD
POLITECNICA
DE VALENCIA

Escola Politècnica Superior de Gandia

DSIC

Departament de Sistemes Informàtics i Computació

Práctica 1

¡ Atención !

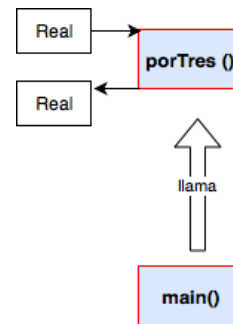
- ▷ Se recuerda que las prácticas deben prepararse antes de acudir al aula informática, anotando en el enunciado las dudas que se tengan.
- ▷ Los diseños y algoritmos que se piden en esta práctica deben escribirse en la libreta de apuntes para poder ser revisados.
- ▷ Utiliza *git* en cada ejercicio haciendo *commit* cada vez que consigas un "hito".
- ▷ La realización de las prácticas es un trabajo individual y original. En caso de plagio se excluirá al alumno de la asignatura. Por tanto, es preferible presentar el trabajo realizado por uno mismo aunque éste tenga errores.



1

Funciones sencillas y "callbacks"

La función `porTres()` recibe un número real y devuelve el resultado de multiplicarlo por tres. Su diseño es el siguiente:



Y su algoritmo es éste:

Datos de entrada: $a : \mathbb{R}$

Datos de salida: \mathbb{R}

Devolver $a \cdot 3$

1. En el fichero `mainPorTres1.js`, implementa el diseño de la función `porTres()` utilizando un `return` para devolver el resultado. Llama a esta función escribiendo una prueba automática para comprobar que va correctamente.

▷ Atención: "devolver" no es escribir en pantalla.

- ▷ Atención: Hay dos tipos mutuamente excluyentes de funciones:
 - las de entrada/salida: se comunican con el usuario (escriben en pantalla y/o leen de teclado). No hacen cálculos. Ejemplo: `main()`.
 - las de cálculo: no se comunican con el usuario (ni leen ni escriben), sólo calculan. Ejemplo: `porTres()`.
- 2. En el fichero `mainPorTres2.js`, implementa el diseño de la función `porTres()` utilizando un `callback` para devolver el resultado. Llama a esta función escribiendo una prueba automática para comprobar que va correctamente.



2

Asincronía

1. La función de biblioteca `setTimeout()` recibe un `callback` y la cantidad de milisegundos que debe esperar antes de llamar al callback.

```
//  
// main ()  
//  
console.log( " antes " )  
  
setTimeout( function() {  
    console.log( " pasaron dos segundos " )  
}, 2000)  
  
console.log( " después " )
```

Escribe el anterior código en un fichero `.js`, Pruébalo y deduce:

¿por qué lo último en escribirse en pantalla es "pasaron dos segundos" y no "después"?

2. Prueba el siguiente código

```
setTimeout ( function () {  
    console.log ( " hola 1 " );  
}, 2000 )  
  
setTimeout ( function () {  
    console.log ( " hola 2 " );  
}, 2000 )
```

Responde: ¿por qué salen los dos mensajes al mismo tiempo?

Esto ocurre ya que el programa se sigue ejecutando mientras espera los dos segundos

Esto ocurre ya que el programa se sigue ejecutando mientras espera los dos segundos



Modifica el anterior código para que el primer mensaje tarde 2 segundos en aparecer y el siguiente mensaje aparezca 2 segundos tras el primero.

3. Escribe una versión del diseño de la función `porTres()` en la que ésta tarde tres segundos en devolver el resultado. Escribe una prueba automática para esta versión de la función.



3

Filtros

```
//  
//  
//  
function sumar( inicio, fin, condicion ) {  
  var total = 0  
  for( var i=inicio; i<=fin; i++ ) {  
    if ( condicion(i) ) {  
      total = total + i  
    }  
  } // for  
  return total  
} // ()  
  
//  
// main()  
//  
var s = sumar( 1, 10, function(e) {  
  if (e % 3 == 0 ) {  
    return true  
  }  
  return false  
})  
  
console.log( s )
```

- ¿Qué hace la función `sumar()`? Suma los numeros del 1 hasta el 10 si cumplen una condicion (en este caso si son divisibles por tres)
- ¿Es asíncrona la función `sumar()`? Es ASÍNCRONA ya que al ejecutarse, llama a la función que comprueba si se cumple la condición (si es divisible por 3 en este caso) y tiene que esperar a que dicha función devuelva un valor para continuar ejecutándose



- ¿Es la función `function(e) { ... }` un callback? Si
- ¿Cuál es el diseño de la función `sumar()`?
- ¿Por qué cuando una función recibe un callback, éste no aparece en el diseño?
- Diseña, implementa y realiza pruebas automáticas para una función que calcule:

$$\int_a^b f(x)dx$$

Porque el callback puede tener diferentes implementaciones (se implementa cuando se llama a la función). Lo único que la función principal necesita saber es lo que devuelve el callback (en este caso un boolean)



30 enero 2019