GUIÓN DE LA ACTIVIDAD AEV6:

Título

Pruebas unitarias

Objetivos

- Entender e identificar las diferentes tipologías de pruebas en el proceso de desarrollo de software.
- Analizar especificaciones de requerimientos para calcular y definir casos de prueba que permitan incrementar el nivel de fiabilidad de un producto software.
- Diseñar y desarrollar pruebas unitarias que permitan el eficaz testeo de fragmentos de código específicos.
- Prepara la automatización de las pruebas unitarias utilizando herramientas de entorno de desarrollo específicas para esta finalidad.

Temporalización

Se estima una dedicación de **5 horas**. Teniendo en cuenta que habrá que revisar los recursos facilitados en el curso en Florida Oberta para realizar la actividad.

Proceso de desarrollo

1. Leer detenidamente y entender cada uno de los pasos propuestos y especificados en el detalle de la actividad.
2. Generar la información necesaria, ya sea literaria o gráfica, para dar respuesta completa a lo solicitado en cada uno de los pasos. Lo más importante será la explicación de los procedimientos y deducciones.
3. Entregar un documento PDF, debidamente identificado, que incluya cada enunciado con la respuesta correspondiente, a través de Florida Oberta.

Evaluación

La actividad consiste en llevar a cabo una serie de pasos que simularán una parte de un plan de pruebas, desde el punto de vista de un equipo de desarrollo de software. Cada uno de los pasos se valorará en función de su dificultad y esfuerzo requerido. En total, los todos los pasos sumarán 10 puntos. Cada error o carencia en alguno de ellos implica un descuento de:

- Entre 0,25 puntos y 2 puntos, si el error es leve.
- Entre 1 punto y el paso completo, si el error es grave o muy grave.

  * Se considera error grave o muy grave a la ausencia o incorrecta expresión de elementos básicos que condicionen de forma severa la corrección de lo presentado, o bien presentar información que directamente no se corresponda con el planteamiento del enunciado. Se considera error o carencia leve, el resto de las incorrecciones.

Recursos

Puestos a disposición del alumno en el curso correspondiente del campus virtual Florida Oberta.

Detalle de la actividad

Supón que formas parte de un equipo de desarrollo de software. Sigue los pasos indicados, en base al fragmento de código que se proporciona, para llevar a cabo una parte de un plan de pruebas que permita testear adecuadamente la función.

La tipología de las pruebas a realizar será el siguiente:

- Según el objetivo: **Pruebas unitarias**.
- Según el enfoque: **Pruebas de caja blanca**.
- Según el método: **Pruebas automáticas.**

```
//Mecanismo de ordenación ascendente
function Ordenacion(secuencia) {

    //Índices i y k ; auxiliar para intercambio
    var i, k, aux;

    //Mostramos por consola la secuencia tal y como llega
    console.log(secuencia);

    //Bucle de ordenación
    for (k = 1; k < secuencia.length; k++) {

        for (i = 0; i < (secuencia.length - k); i++) {

            if (secuencia[i] > secuencia[i + 1]) {

                //Intercambio entre la posición i y la posterior
                aux = secuencia[i];
                secuencia[i] = secuencia[i + 1];
                secuencia[i + 1] = aux;
            }
        }
    }

    // Mostramos por consola la secuencia  ordenada
    console.log(secuencia);
    return secuencia;
}

//Llamada de ejemplo a la función (el ejemplo puede variar, claro…)
var ejemplo = [47, 28, 1, 32, 95, 6, 54, 73, 19, 0]
resultado = Ordenacion(ejemplo);
```

1. Vamos a utilizar la técnica de la **Prueba del camino básico**, por lo tanto, lo primero que hay que hacer es diseñar el grafo de flujo de control y calcular la complejidad ciclomática de la función del fragmento de código facilitado. Detalla para cada nodo del grafo, cuál sería la funcionalidad que contiene o encapsula. **(3 puntos)**

2. Una vez obtenida la complejidad ciclomática, redacta de forma completa los **Casos de prueba** que consideres convenientes para probar la función del fragmento de código ofreciendo un nivel de garantía mínimo aceptable. **(3 puntos)**

3. Instala una herramienta para gestionar la **automatización** de las pruebas unitarias (se propone el uso de Jest). Desarrolla los scripts de prueba unitaria correspondientes a los casos de prueba definidos y ejecuta las pruebas unitarias. **(3 puntos)**

4. Redacta un breve **Informe de pruebas** que resuma el resultado de las pruebas unitarias aplicadas. **(1 puntos)**