Programa 6

Proceso Personal para el Desarrollo de Software

Este material fue realizado en base a material del curso “Personal Software Process for Engineers: PartI”, dictado por The Software Engineering Institute (SEI)

Proceso Personal para el Desarrollo de Software

Instructivo para el Programa 6

Descripción

Descripción

El presente instructivo cubre los siguientes temas

Programa 6 Requerimientos

Requerimientos Programa 6

Utilizando PSP0.1, escribir un programa para encontrar el valor de *x* para el cual integrando la función t desde 0 hasta *x* se obtiene el resultado *p*

Probar a fondo el programa. Como mínimo pruebe el sistema para los valores de la Tabla 1. Los valores esperados se incluyen también en la tabla 1.

Test		Expected Value	Actual Value
p	***dof***	***x***	
0.20	6	0.55338	
0.45	15	1.75305	
0.495	4	4.60409	

Table 1

Algoritmo de búsqueda

Buscar el valor de *x*

Encontrar el valor de *x* para el cual integrar la función t entre 0 y *x* da el resultado *p*.

- Comience con un valor de prueba 1 para el limite superior y calcule el valor de la integral.
- Compárelo con el valor deseado.
  - Si el resultado de la integración es menor, seleccione y pruebe un valor mayor.
  - Si el resultado de la integración es mayor, seleccione y pruebe un valor menor.

Realice sucesivas pruebas hasta que el valor de la integración se encuentre dentro del valor aceptable de error, 0.00001.

Una forma de realizar estos cálculos es la siguiente

Paso	***Acción***
1	Comenzar con un valor de prueba de x (por ejemplo, 1.0).
2.	Calcular una integral inicial y probar para ver si da el valor apropiado; si no, continuar.
3.	Si es demasiado bajo, agregar *d* = 0.5 al valor de prueba x
4.	Si es demasiado alto, restar *d* = 0.5 al valor de prueba x
5.	Integrar otra vez y probar si el resultado está dentro de un error aceptable; si no, continuar
6.	Si es demasiado bajo, ajustar *d*; agregar *d* al valor de prueba x
7.	Si es demasiado alto, ajustar *d*; restar *d* al valor de prueba x
8.	Volver al punto 5.

Las reglas para ajustar *d* son las siguientes.

1. Mientras las pruebas para el error del resultado den el mismo signo del error, dejar *d* sin cambios.
2. Cuando el signo del error cambie, divida *d* entre 2.

Observar que este método de ajustar *d* podría dar lugar a un valor de prueba x = 0.

Para evitar un problema con el método de Simpson, asegúrese que el programa manejará el valor 0 de la función que es integrada.

Instrucciones

Instrucciones

Antes de comenzar el programa 6, repasar el proceso PSP0.1 para asegurarse de comprenderlo. También, asegurarse de tener todas las entradas requeridas antes de comenzar con la fase de planificación.

Entregables

Cuando complete la etapa de postmortem, arme un archivo .zip para enviar al instructor, conteniendo lo siguiente.

- El archivo .mdb con sus datos del ejercicio.
- Código fuente del programa.
- Impresión de pantalla de las pruebas realizadas mostrando el resultado de la mismas.
- Captura de pantalla de la salida del contador de LOC aplicada al ejercicio 6.
- Archivo de texto o documento que contenga la descripción de que clases/módulos/unidades del código construido contienen las distintas categorías (Added, Modified, Delete, etc). Utilizando el contador de LOC que muestra el tamaño de dichas unidades de código.

Ejemplo:

36 LOC Base -> Clase Base.java y muestre el tamaño de la clase utilizando el contador de LOCS aplicado a la clase.

3 LOC Deleted -> En la clase Base.java

10 LOC Modified -> En la clase Base.java

15 LOC Added -> En la clase Base.java

40 LOC Reused -> Clases LibUno.java, LibDos.java y muestre el tamaño de la las mismas utilizando el contador de LOCS aplicado ambas clases.

5 LOC New Reused -> agregadas en NuevaLib.java

Tenga en cuenta que:

.- Deben programar de acuerdo a su estándar de codificación.

.- Deben utilizar el mismo proceso que en el ejercicio anterior, PSP 0.1.
Esto quiere decir que en planificación van a estimar la cantidad de LOC que esperan generar y si parten de algún LOC Base van a indicar cuántas líneas son. Luego, en postmortem, van a usar el contador de LOC que construyeron para contar la cantidad de LOC del ejercicio 6.

.- En este ejercicio puede ocurrir alguna diferencia entre los resultados de su programa y los resultados esperados, debido a las aproximaciones.

Si utiliza el error propuesto por el ejercicio, error = 0.00001, puede ocurrir alguna diferencia con el valor esperado en los tests. Puede probar achicando el error y utilizar error=0.000001.

Si las diferencias que obtiene por las aproximaciones son razonables, se entiende que el ejercicio está resuelto correctamente y las diferencias se deben a las funciones propias del lenguaje.

Criterios de evaluación para el programa 6

Criterios de Evaluación

Su reporte debe ser
- completo

Los datos de su proceso deben ser
- exactos
- precisos
- consistentes

Sugerencias

Recuerde entregar su reporte en fecha.

Mantenga la simplicidad del programa.

Si no está seguro de algo, consulte a su instructor

Debe producir y reportar sus propias estimaciones y datos reales, desarrollar su propio código y llevar adelante su propio juego de pruebas.

