|  |
| --- |
| # tarea6  # git url: https://github.com/axxa/tarea6.git  # heroku url: http://tarea6-alvarosuarez.herokuapp.com |

PSP2.0 Project Plan Summary

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Alvaro Andres Suarez Alfonso | Date | 01 Mar 2015 |
| Program | Tarea 6 | Program # | CSOF5101\_01\_6 |
| Instructor | Luis Daniel Benavides Navarro | Language |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Summary*** | ***Plan*** | | | |  | ***Actual*** | | |  | ***To Date*** | | |
| ***Size/Hour*** | *180* | | | |  | *0.8* | | |  | *90.4* | | |
| ***Planned Time*** | 100 | | | |  |  | | |  | 328 | | |
| ***Actual Time*** |  | | | |  | 679 | | |  | 907 | | |
| ***CPI (Cost-Performance Index)*** | | | | |  |  | | |  | 0.36 | | |
|  |  | | | |  |  | | |  | (Planned/Actual) | | |
| Test Defects/KLOC or equivalent | | (1000\*2)/50= 40 | | |  | (1000\*1)/153=6.53 | | |  | 19.18 | | |
| Total Defects/KLOC or equivalent | | (1000\*3)/260= 11.53 | | |  | (1000\*3)/237=12.65 | | |  | 12.65 | | |
| Yield % | | 100% | | |  | 100% | | |  | 100% | | |
| ***% Reused*** | 60% | | | |  | 52.26% | | |  | 63.73% | | |
| ***% New Reusable*** | 60% | | | |  | 75.21% | | |  | 75.21% | | |
|  |  | | | |  |  | | |  |  | | |
| ***Program Size*** | ***Plan*** | | | |  | ***Actual*** | | |  | ***To Date*** | | |
| ***Base (B)*** |  | | | |  | 230 | | |  |  | | |
|  |  | | | |  | ***(Measured)*** | | |  |  | | |
| ***Deleted (D)*** |  | | | |  | 34 | | |  |  | | |
|  |  | | | |  | ***(Counted)*** | | |  |  | | |
| ***Modified (M)*** |  | | | |  | 5 | | |  |  | | |
|  |  | | | |  | ***(Counted)*** | | |  |  | | |
| ***Added (A)*** |  | | | |  | 148 | | |  |  | | |
|  |  | | | |  | ***(T − B + D − R)*** | | |  |  | | |
| ***Reused (R)*** |  | | | |  | 196 | | |  | 1213 | | |
|  |  | | | |  | ***(Counted)*** | | |  |  | | |
| ***Added and Modified (A+M)*** | 50 | | | |  | 153 | | |  | 1346 | | |
|  |  | | | |  | ***(A + M)*** | | |  |  | | |
| ***Total Size (T)*** |  | | | |  | 375 | | |  | 1560 | | |
|  |  | | | |  | ***(Measured)*** | | |  |  | | |
| ***Total New Reusable*** |  | | | |  | 196 | | |  | 1213 | | |
|  |  | | | |  |  | | |  |  | | |
| **Time in Phase (min.)** | ***Plan*** | |  | **Actual** | | |  | **To Date** | | |  | **To Date %** |
| Planning | 10 | |  | 6 | | |  | 75 | | |  | 4.26% |
| Design | 15 | |  | 4 | | |  | 48 | | |  | 2.73% |
| Code | 200 | |  | 613 | | |  | 1408 | | |  | 80.09% |
| Compile | 0 | |  | 0 | | |  | 49 | | |  | 2.78% |
| Test | 10 | |  | 46 | | |  | 146 | | |  | 8.30% |
| Postmortem | 10 | |  | 10 | | |  | 132 | | |  | 7.50% |
| Total | 200 | |  | 679 | | |  | 1758 | | |  | 100% |
|  |  | |  |  | | |  |  | | |  |  |
| **Defects Injected** |  | |  | **Actual** | | |  | **To Date** | | |  | **To Date %** |
| Planning |  | |  | 0 | | |  | 0 | | |  | 0% |
| Design |  | |  | 0 | | |  | 0 | | |  | 0% |
| Code |  | |  | 1 | | |  | 14 | | |  | 100% |
| Compile |  | |  | 0 | | |  | 0 | | |  | 0% |
| Test |  | |  | 0 | | |  | 0 | | |  | 0% |
| Total Development |  | |  | 1 | | |  | 14 | | |  | 100% |
|  |  | |  |  | | |  |  | | |  |  |
| **Defects Removed** |  | |  | **Actual** | | |  | **To Date** | | |  | **To Date %** |
| Planning |  | |  | 0 | | |  | 0 | | |  | 0% |
| Design |  | |  | 0 | | |  | 0 | | |  | 0% |
| Code |  | |  | 0 | | |  | 7 | | |  | 50% |
| Compile |  | |  | 0 | | |  | 0 | | |  | 0% |
| Test |  | |  | 1 | | |  | 7 | | |  | 50% |
| Total Development |  | |  | 1 | | |  | 14 | | |  | 100% |
| After Development |  | |  | 0 | | |  | 0 | | |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***Defect Removal Efficiency*** | ***Plan*** |  | ***Actual*** |  | ***To Date*** |
| ***Defects/Hour − Design Review*** | 0 |  | 0 |  | 0 |
| ***Defects/Hour − Code Review*** | 0 |  | 0 |  | 0 |
| ***Defects/Hour − Compile*** | 0 |  | 0 |  | 0 |
| ***Defects/Hour − Test*** | 30 |  | 0.5 |  | 2.5 |
| ***DRL (DLDR/UT)*** |  |  |  |  |  |
| ***DRL (Code Review/UT)*** |  |  |  |  |  |
| ***DRL (Compile/UT)*** |  |  |  |  |  |

PSP Time Recording Log

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Alvaro Suarez | Date | 01 Mar 2015 |
| Program | Tarea 6 | Program # | CSOF5101\_01\_6 |
| Instructor | Luis Daniel Benavides Navarro | Language | Java |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Project** | **Phase** | **Start Date and Time** | **Int. Time** | **Stop Date and Time** | **Delta**  **Time** | **Comments** |
| T6-01\_mar | Plan | 8.47 |  | 8.53 | 6 |  |
|  | Desi | 8.53 |  | 8.57 | 4 |  |
|  | Code | 8.58 |  | 9.32 | 34 |  |
|  | Code | 9.57 |  | 12.26 | 149 |  |
|  | Code | 14.20 | 50 | 21.44 | 480 |  |
|  | Test | 21.44 |  | 22.30 | 46 |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

PSP Defect Recording Log

|  |  |
| --- | --- |
| Defect Types |  |
| 10 Documentation | 60 Checking |
| 20 Syntax | 70 Data |
| 30 Build, Package | 80 Function |
| 40 Assignment | 90 System |
| 50 Interface | 100 Environment |

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Alvaro Suarez | Date | 01 Mar 2015 |
| Program | Tarea 6 | Program # | CSOF5101\_01\_6 |
| Instructor | Luis Daniel Benavides Navarro | Language | Java |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
| Tarea 6 |  | | 01 Mar 9.57 am |  | 1 |  | Calculo |  | code |  | test |  | 22.20pm |  |  |
| Description: | | | Calculando x final | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Description: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Description: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Description: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |

Source programs listing

Clase Main:

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.eclipse.jetty.server.Server;

import org.eclipse.jetty.servlet.ServletContextHandler;

import org.eclipse.jetty.servlet.ServletHolder;

/\*\*

\* Clase Main

\* @author alvaro suarez

\* @version 1.0.20150225

\* @since 1.0.20150225

\*/

public class Main extends HttpServlet{

/\*\*

\* doGet

\* @param req HttpServletRequest

\* @param resp HttpServletResponse

\* @throws ServletException excepcion

\* @throws IOException excepcion

\*/

@Override

protected void doGet(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException {

showHome(req,resp);

}

/\*\*

\* Prepara los datos a mostrar

\* @param req HttpServletRequest

\* @param resp HttpServletResponse

\* @throws IOException excepcion

\*/

private void showHome(HttpServletRequest req, HttpServletResponse resp) throws IOException{

show("1",new XCalculador(1),req,resp);

show("2",new XCalculador(2),req,resp);

show("3",new XCalculador(3),req,resp);

}

/\*\*

\* Muestra los datos

\* @param testName String

\* @param cd XCalculador

\* @param req HttpServletRequest

\* @param resp HttpServletResponse

\* @throws IOException excepcion

\*/

private void show(String testName,XCalculador cd,HttpServletRequest req, HttpServletResponse resp) throws IOException{

resp.getWriter().print("\n \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

resp.getWriter().print("\n TEST " + testName);

resp.getWriter().print("\n x: " + cd.getLimiteF());

//resp.getWriter().print("\n M: " + cd.getM() + " L: " + cd.getL());

//resp.getWriter().print("\n VL: " + cd.getvL());

resp.getWriter().print("\n \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

}

/\*\*

\* Metodo main

\* @param args String[]

\* @throws Exception excepcion

\*/

public static void main(String[] args) throws Exception {

Server server = new Server(Integer.valueOf(System.getenv("PORT")));

ServletContextHandler context = new ServletContextHandler(ServletContextHandler.SESSIONS);

context.setContextPath("/");

server.setHandler(context);

context.addServlet(new ServletHolder(new Main()),"/\*");

server.start();

server.join();

}

}

Clase: Distribution

import java.text.DecimalFormat;

import java.text.ParseException;

import java.util.LinkedList;

import java.util.List;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\* Clase Distribution guarda los parametros base para la funcion de simpson

\* @author alvaro suarez

\* @version 1.0.20150225

\* @since 1.0.20150225

\*/

public class Distribution {

private double limitei;

private double limitef;

private double dof;

private double w;

private double num\_seg;

/\*\*

\* Constructor

\* @param limitei double

\* @param limitef double

\* @param dof double

\* @param num\_seg double

\*/

public Distribution(double limitei,double limitef,double dof, double num\_seg){

this.limitei = limitei;

this.limitef = limitef;

this.dof = dof;

this.num\_seg = num\_seg;

try {

this.w = truncDouble(limitef / num\_seg);

} catch (ParseException ex) {

Logger.getLogger(Distribution.class.getName()).log(Level.SEVERE, null, ex);

}

}

/\*\*

\* Constructor generico

\*/

public Distribution(){

super();

}

/\*\*

\* Trunca un double en el formato ###0.##

\* @param d double

\* @return double d

\* @throws ParseException excepcion

\*/

public double truncDouble(double d) throws ParseException{

DecimalFormat formateador = new DecimalFormat("###0.##");

return formateador.parse(formateador.format(d)).doubleValue();

}

/\*\*

\* @return the limitei

\*/

public double getLimitei() {

return limitei;

}

/\*\*

\* @return the limitef

\*/

public double getLimitef() {

return limitef;

}

/\*\*

\* @return the dof

\*/

public double getDof() {

return dof;

}

/\*\*

\* @return the w

\*/

public double getW() {

return w;

}

/\*\*

\* @return the num\_seg

\*/

public double getNum\_seg() {

return num\_seg;

}

}

Clase: TablaDatos

import java.text.DecimalFormat;

import java.text.ParseException;

import java.util.LinkedList;

import java.util.List;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\* Clase TablaDatos tiene los datos de prueba y los calcula

\* @author alvaro suarez

\* @version 1.0.20150225

\* @since 1.0.20150225

\*/

public class TablaDatos {

public static double ACCEPTABLEERROR = 0.00001;

public static double NUM\_SEG = 10;

public static double LIMITEI = 0;

private List<TablaDatos> tablaDatos;

private Distribution d;

private int i;

private double xi;

private double fX;

private int multiplier;

private double terms;

private double p;

/\*\*

\*

\* @param limitei double

\* @param limitef double

\* @param dof double

\* @param num\_seg double

\*/

public TablaDatos(double limitei,double limitef,double dof, double num\_seg){

p = 0;

llenarTabladatos1(limitei,limitef,dof,num\_seg);

}

/\*\*

\* Constructor de fila para TablaDatos

\* @param i int

\* @param xi double

\* @param multiplier int

\*/

private TablaDatos(int i,double xi,int multiplier){

this.i = i;

this.xi = xi;

this.multiplier = multiplier;

}

/\*\*

\* Llena la tabla con los datos a buscar

\* @param limitei double

\* @param limitef double

\* @param dof double

\* @param num\_seg double

\*/

private void llenarTabladatos1(double limitei,double limitef,double dof, double num\_seg){

this.d = new Distribution(limitei,limitef,dof,num\_seg);

this.tablaDatos = new LinkedList<TablaDatos>();

double aux = getD().getLimitei();

int mult;

double w = getD().getW();

getTablaDatos().add(new TablaDatos(0,aux,1));

for(int j = 1 ; j < getD().getNum\_seg() + 1 ;j++ ){

aux = aux + w;

mult = esPar(j);//multiplier(aux);

if(j == getD().getNum\_seg())

mult = 1;

getTablaDatos().add(new TablaDatos(j,aux,mult));

}

calculadas();

}

/\*\*

\* Calcula la integral

\*/

private void calculadas(){

double exp = (getD().getDof() + 1)/2;

double rightside;

double numeradorrightside;

double denominadorrightside;

double leftside;

for(int i = 0; i<getTablaDatos().size();i++){

try {

fX = 1 + Math.pow(getTablaDatos().get(i).xi, 2)/getD().getDof();

fX = truncDouble(Math.pow(getfX(), -exp));

numeradorrightside =la\_gamma(exp);

denominadorrightside = Math.pow(getD().getDof()\* Math.PI, 0.5)\*la\_gamma(getD().getDof()/2);

rightside = numeradorrightside/denominadorrightside;

fX = rightside \* getfX();

terms = (getD().getW()/3) \* getTablaDatos().get(i).multiplier \* getfX();

fX = truncDouble(getfX());

terms = truncDouble(getTerms());

p = p + terms;

//System.out.println("\n fx: " + fX);

//System.out.println("\n terms: " + terms);

} catch (ParseException ex) {

Logger.getLogger(TablaDatos.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

/\*\*

\* Retorna 2 si es par en caso contrario retorn 4

\* @param n int

\* @return int

\*/

private int esPar(int n){

if(n % 2 == 0)

return 2;

else

return 4;

}

/\*\*

\* Trunca un double en el formato ###0.#####

\* @param d double

\* @return double d

\* @throws ParseException

\*/

public double truncDouble(double d) throws ParseException{

DecimalFormat formateador = new DecimalFormat("###0.#####");

return formateador.parse(formateador.format(d)).doubleValue();

}

/\*\*

\* factorial de un entero

\* @param d double

\* @return double

\*/

private double factorial(double d){

if (d <= 1)

return 1;

else

return d \* factorial(d - 1);

}

/\*\*

\* factorial de un fraccionario o entero

\* @param x double

\* @return double

\*/

public double la\_gamma(double x){

if(x % 1 == 0){

return factorial(x-1);

}

else{

double[] p = {0.99999999999980993, 676.5203681218851, -1259.1392167224028,

771.32342877765313, -176.61502916214059, 12.507343278686905,

-0.13857109526572012, 9.9843695780195716e-6, 1.5056327351493116e-7};

int g = 7;

if(x < 0.5) return Math.PI / (Math.sin(Math.PI \* x)\*la\_gamma(1-x));

x -= 1;

double a = p[0];

double t = x+g+0.5;

for(int i = 1; i < p.length; i++){

a += p[i]/(x+i);

}

return Math.sqrt(2\*Math.PI)\*Math.pow(t, x+0.5)\*Math.exp(-t)\*a;

}

}

public double getfX() {

return fX;

}

public double getTerms() {

return terms;

}

public double getP() {

return p;

}

public List<TablaDatos> getTablaDatos() {

return tablaDatos;

}

public Distribution getD() {

return d;

}

}

Clase: XCalculador

/\*\*

\* Clase XCalculador tiene los datos de prueba y los calcula

\* @author alvaro suarez

\* @version 1.0.20150228

\* @since 1.0.20150225

\*/

public class XCalculador {

public static double ACCEPTABLEERROR = 0.015;

public static double NUM\_SEG = 10;

public static double LIMITEI = 0;

private double limiteF;

private double p;

/\*\*

\* Constructor

\* @param mostrar int

\*/

public XCalculador(int mostrar){

TablaDatos td;

double limitef = 0.1;

double dof=0;

double pToGet=0;

if(mostrar == 1){

dof = 6;

pToGet = 0.20;

}

if(mostrar == 2){

dof = 15;

pToGet = 0.45;

}

if(mostrar == 3){

dof = 4;

pToGet = 0.495;

}

findX(limitef, dof, pToGet);

}

/\*\*

\* busca el valor x

\* @param limitef double

\* @param dof double

\* @param pToGet double

\*/

private void findX(double limitef,double dof,double pToGet){

TablaDatos td;

do{

td = new TablaDatos(LIMITEI,limitef,dof,NUM\_SEG);

limitef = limitef+0.001;

}while(!(td.getP() > pToGet && td.getP()<pToGet+ACCEPTABLEERROR)) ;

this.limiteF = td.getD().getLimitef();

this.p = td.getP();

}

public double getLimiteF() {

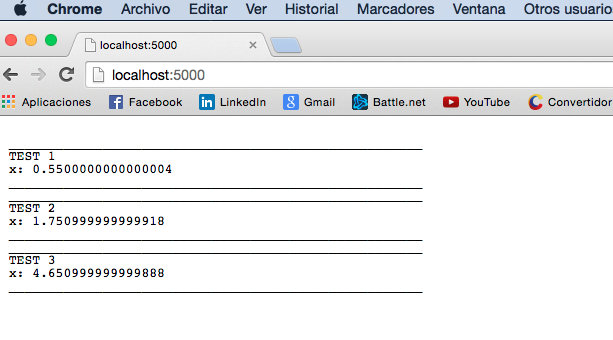
return limiteF;

}

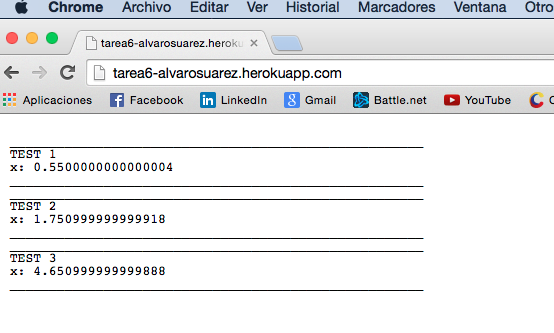
}

Test results

Pruebas Locales:

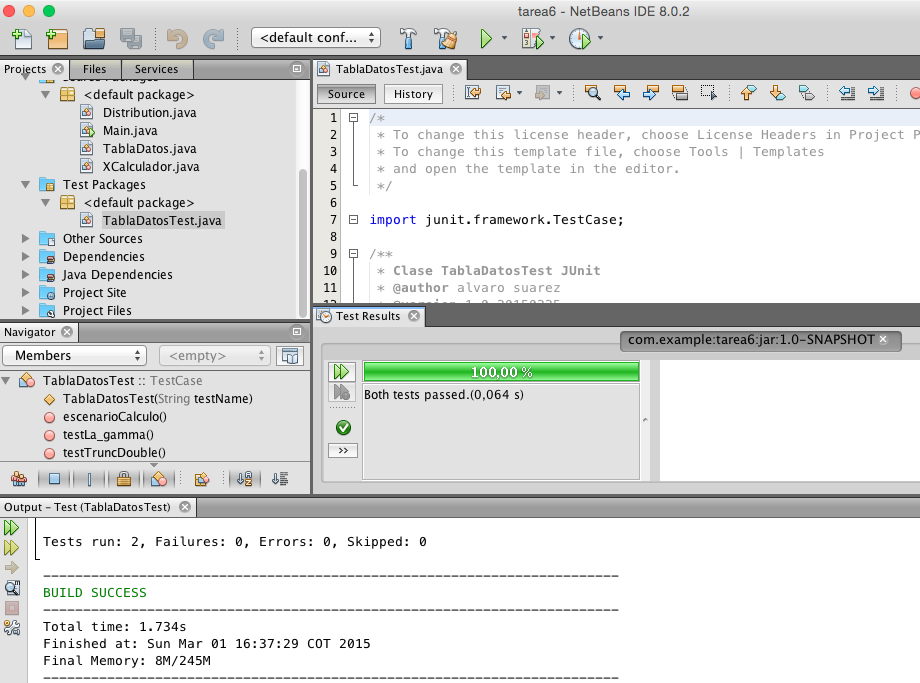


Pruebas en la nube:



Pruebas Unitarias:

TablaDatosTest



* Clase: TablaDatosTest

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

import junit.framework.TestCase;

/\*\*

\* Clase TablaDatosTest JUnit

\* @author alvaro suarez

\* @version 1.0.20150225

\* @since 1.0.20150225

\*/

public class TablaDatosTest extends TestCase {

public TablaDatosTest(String testName) {

super(testName);

}

/\*\*

\* Test of truncDouble method, of class TablaDatos.

\*/

public void testTruncDouble() throws Exception {

System.out.println("truncDouble");

double d = 1.123456789;

TablaDatos instance = new TablaDatos(0,0.55338,6,10);

double expResult = 1.12346;

double result = instance.truncDouble(d);

System.out.println(result);

assertEquals(expResult, result);

}

/\*\*

\* Test of la\_gamma method, of class TablaDatos.

\*/

public void testLa\_gamma() {

System.out.println("la\_gamma");

double x = 4.5;

TablaDatos instance = new TablaDatos(0,0.55338,6,10);

double expResult = 11.631728396567446;

double result = instance.la\_gamma(x);

assertEquals(expResult, result);

}

/\*\*

\* Test de calculo de TablaDatos.

\*/

public void escenarioCalculo() {

System.out.println("escenarioCalculo");

double incertidumbre = 0.01;

TablaDatos instance = new TablaDatos(0,0.55338,6,10);

double expResult = 0.35006;

double result = instance.getP();

assertTrue("fail",result>expResult && result<expResult+incertidumbre);

}

}