|  |  |
| --- | --- |
| # tarea3  # git url: https://github.com/axxa/tarea3.git  # heroku url: http://tarea3-alvarosuarez.herokuapp.com  PSP0.1 Development Script | |
| Purpose | To guide the development of small programs |
| Entry Criteria | * Requirements statement * Project Plan Summary form with estimated program ***size and*** development time * Time and Defect Recording logs * Defect Type standard ***and Coding standard*** |

|  |  |  |
| --- | --- | --- |
| Step | Activities | Description |
| 1 | Design | * Review the requirements and produce a design to meet them. * Record in the Defect Recording log any requirements defects found. * Record time in the Time Recording log. |
| 2 | Code | * Implement the design ***following the Coding standard.*** * Record in the Defect Recording log any requirements or design defects found. * Record time in the Time Recording log. |
| 3 | Compile | * Compile the program until there are no compile errors. * Fix all defects found. * Record defects in the Defect Recording log. * Record time in the Time Recording log. |
| 4 | Test | * Test until all tests run without error. * Fix all defects found. * Record defects in the Defect Recording log. * Record time in the Time Recording log. |

|  |  |
| --- | --- |
| Exit Criteria | * A thoroughly tested program ***that conforms to the Coding standard*** * Completed Time and Defect Recording logs |

|  |  |
| --- | --- |
| PSP0.1 Postmortem Script | |
| Purpose | To guide the PSP postmortem process |
| Entry Criteria | * Problem description and requirements statement * Project Plan Summary form with program size and development timedata * Completed Time and Defect Recording logs * A tested and running program ***that conforms to the coding and size measurement standards*** |

|  |  |  |
| --- | --- | --- |
| Step | Activities | Description |
| 1 | Defect Recording | * Review the Project Plan Summary to verify that all of the defects found in each phase were recorded. * Using your best recollection, record any omitted defects. |
| 2 | Defect Data Consistency | * Check that the data on every defect in the Defect Recording log are accurate and complete. * Verify that the numbers of defects injected and removed per phase are reasonable and correct. * Using your best recollection, correct any missing or incorrect defect data. |
| ***3*** | ***Size*** | * ***Count the size of the completed program.*** * ***Determine the size of the base, reused, deleted, modified, added, total, added and modified, and new reusable code.*** * ***Enter these data in the Project Plan Summary form.*** |
| 4 | Time | * Review the completed Time Recording log for errors or omissions. * Using your best recollection, correct any missing or incomplete time data. |

|  |  |
| --- | --- |
| Exit Criteria | * A thoroughly tested program ***that conforms to the coding and size measurement standards*** * Completed Project Plan Summary form * ***Completed PIP forms describing process problems, improvement suggestions, and lessons learned*** * Completed Time and Defect Recording logs |

PSP1 Project Plan Summary

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Alvaro Andres Suarez Alfonso | Date | 9 Feb 2015 |
| Program | Tarea 3 | Program # | CSOF5101\_01\_3 |
| Instructor | Luis Daniel Benavides Navarro | Language |  |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Summary*** | ***Plan*** | | |  | ***Actual*** | | |  | ***To Date*** | | |
| ***Size/Hour*** | *200/1* | | |  | *230/1* | | |  | *230/1* | | |
|  |  | | |  |  | | |  |  | | |
| ***Program Size*** | ***Plan*** | | |  | ***Actual*** | | |  | ***To Date*** | | |
| ***Base (B)*** |  | | |  | 0 | | |  |  | | |
|  |  | | |  | ***(Measured)*** | | |  |  | | |
| ***Deleted (D)*** |  | | |  | 0 | | |  |  | | |
|  |  | | |  | ***(Counted)*** | | |  |  | | |
| ***Modified (M)*** |  | | |  | 0 | | |  |  | | |
|  |  | | |  | ***(Counted)*** | | |  |  | | |
| ***Added (A)*** |  | | |  | 314 | | |  |  | | |
|  |  | | |  | ***(T − B + D − R)*** | | |  |  | | |
| ***Reused (R)*** |  | | |  | 440 | | |  | 613 | | |
|  |  | | |  | ***(Counted)*** | | |  |  | | |
| ***Added and Modified (A+M)*** | 260 | | |  | 314 | | |  | 728 | | |
|  |  | | |  | ***(A + M)*** | | |  |  | | |
| ***Total Size (T)*** |  | | |  | 314 | | |  | 728 | | |
|  |  | | |  | ***(Measured)*** | | |  |  | | |
| ***Total New Reusable*** |  | | |  | 440 | | |  | 613 | | |
|  |  | | |  |  | | |  |  | | |
| **Time in Phase (min.)** | ***Plan*** |  | **Actual** | | |  | **To Date** | | |  | **To Date %** |
| Planning | 10 |  | 10 | | |  | 24 | | |  | 3.79% |
| Design | 10 |  | 8 | | |  | 31 | | |  | 4.89% |
| Code | 120 |  | 108 | | |  | 498 | | |  | 78.67% |
| Compile | 10 |  | 9 | | |  | 41 | | |  | 6.47% |
| Test | 10 |  | 17 | | |  | 37 | | |  | 5.84% |
| Postmortem | 20 |  | 43 | | |  | 102 | | |  | 16.11% |
| Total | 180 |  | 195 | | |  | 633 | | |  | 100% |
|  |  |  |  | | |  |  | | |  |  |
| **Defects Injected** |  |  | **Actual** | | |  | **To Date** | | |  | **To Date %** |
| Planning |  |  | 0 | | |  | 0 | | |  | 0% |
| Design |  |  | 0 | | |  | 0 | | |  | 0% |
| Code |  |  | 2 | | |  | 9 | | |  | 100% |
| Compile |  |  | 0 | | |  | 0 | | |  | 0% |
| Test |  |  | 0 | | |  | 0 | | |  | 0% |
| Total Development |  |  | 2 | | |  | 9 | | |  | 100% |
|  |  |  |  | | |  |  | | |  |  |
| **Defects Removed** |  |  | **Actual** | | |  | **To Date** | | |  | **To Date %** |
| Planning |  |  | 0 | | |  | 0 | | |  | 0% |
| Design |  |  | 0 | | |  | 0 | | |  | 0% |
| Code |  |  | 0 | | |  | 7 | | |  | 77.77% |
| Compile |  |  | 0 | | |  | 0 | | |  | 0% |
| Test |  |  | 2 | | |  | 2 | | |  | 22.22% |
| Total Development |  |  | 2 | | |  | 9 | | |  | 100% |
| After Development |  |  | 0 | | |  | 0 | | |  |  |

|  |  |  |
| --- | --- | --- |
| PSP0.1 Plan Summary Instructions | |  |
| Purpose | To hold the plan and actual data for programs or program parts | |
| General | * ***Use the most appropriate size measure, either LOC or element count.*** * “To Date” is the total actual to-date values for all products developed. | |
| Header | * Enter your name and the date. * Enter the program name and number. * Enter the instructor’s name and the programming language you are using. | |
| *Program Size* | * ***Enter the plan added and modified size value (A+M).*** * ***Enter actual base, deleted, modified, reused, total, and new reusable size.*** * ***Calculate actual added size as T-B+D-R and actual added and modified size as A+M.*** * ***Enter to-date reused, added and modified, total, and new reusable size.*** | |
| Time in Phase | * Enter the estimated total development time. * ***Distribute the estimated total time across the development phases according to the To Date % for the most recently developed program.*** * Enter the actual time by phase and the total time. * To Date: Enter the sum of the actual times for this program plus the to-date times from the most recently developed program. * To Date %: Enter the percentage of to-date time in each phase. | |
| Defects Injected | * Enter the actual defects by phase and the total actual defects. * To Date: Enter the sum of the actual defects injected by phase and the to-date values for the most recent previously developed program. * To Date %: Enter the percentage of the to-date defects injected by phase. | |
| Defects Removed | * To Date: enter the actual defects removed by phase plus the to-date values for the most recent previously developed program. * To Date %: Enter the percentage of the to-date defects removed by phase. * After development, record any defects subsequently found during program testing, use, reuse, or modification. | |

PSP Time Recording Log

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Alvaro Suarez | Date | 9 Feb 2015 |
| Program | Tarea 3 | Program # | CSOF5101\_01\_3 |
| Instructor | Luis Daniel Benavides Navarro | Language | Java |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Project** | **Phase** | **Start Date and Time** | **Int. Time** | **Stop Date and Time** | **Delta**  **Time** | **Comments** |
| T3-07\_feb | Plan | 9:12 |  | 9:22 | 10 |  |
|  | Desi | 9:22 |  | 9:30 | 8 |  |
|  | Code | 9:45 |  | 11:33 | 108 |  |
|  | Com | 14:01 |  | 14:10 | 9 |  |
|  | Test | 14:19 |  | 14:36 | 17 |  |
|  | Post | 15:00 |  | 15:43 | 43 |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Time Recording Log Instructions

|  |  |
| --- | --- |
| Purpose | * Use this form to record the time you spend on each project activity. * For the PSP, phases often have only one activity; larger projects usually have multiple activities in a single process phase. * These data are used to complete the Project Plan Summary. * Keep separate logs for each program. |
| General | * Record all of the time you spend on the project. * Record the time in minutes. * Be as accurate as possible. * If you need additional space, use another copy of the form. * If you forget to record the starting, stopping, or interruption time for an activity, promptly enter your best estimate. |
| Header | * Enter your name and the date. * Enter the program name and number. * Enter the instructor’s name and the programming language you are using. |
| Project | Enter the program name or number. |
| Phase | Enter the name of the phase for the activity you worked on, e.g. Planning, Design, Test. |
| Start Date and Time | Enter the date and time when you start working on a process activity. |
| Interruption Time | * Record any interruption time that was not spent on the process activity. * If you have several interruptions, enter their total time. * You may enter the reason for the interrupt in comments. |
| Stop Date and Time | Enter the date and time when you stop working on that process activity. |
| Delta Time | Enter the clock time you actually spent working on the process activity, less the interruption time. |
| Comments | Enter any other pertinent comments that might later remind you of any unusual circumstances regarding this activity. |

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 | 9 Feb 2015 |
| Program | Tarea 3 | Program # | CSOF5101\_01\_3 |
| Instructor | Luis Daniel Benavides Navarro | Language | Java |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
| Tarea 3 |  | | 9 Feb |  | 1 |  | Calculo |  | code |  | test |  | 6 |  |  |
| Description: | | | Calculando beta 1 | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
| Tarea 3 |  | | 9 Feb |  | 2 |  | Calculo |  | code |  | test |  | 8 |  |  |
| Description: | | | Calculando rxy | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Description: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Description: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |

PSP Defect Recording Log Instructions

|  |  |
| --- | --- |
| Purpose | * Use this form to hold data on the defects that you find and correct. * These data are used to complete the Project Plan Summary form. |
| General | * Record each defect separately and completely. * If you need additional space, use another copy of the form. |
| Header | * Enter your name and the date. * Enter the program name and number. * Enter the instructor’s name and the programming language you are using. |
| Project | * Give each program a different name or number. * For example, record test program defects against the test program. |
| Date | Enter the date on which you found the defect. |
| Number | * Enter the defect number. * For each program or module, use a sequential number starting with 1 (or 001, etc.). |
| Type | * Enter the defect type from the defect type list summarized in the top left corner of the form. * Use your best judgment in selecting which type applies. |
| Inject | * Enter the phase when this defect was injected. * Use your best judgment. |
| Remove | Enter the phase during which you fixed the defect. (This will generally be the phase when you found the defect.) |
| Fix Time | * Enter the time that you took to find and fix the defect. * This time can be determined by stopwatch or by judgment. |
| Fix Ref. | * If you or someone else injected this defect while fixing another defect, record the number of the improperly fixed defect. * If you cannot identify the defect number, enter an X. |
| Description | Write a succinct description of the defect that is clear enough to later remind you about the error and help you to remember why you made it. |

PSP Defect Type Standard

|  |  |  |
| --- | --- | --- |
| **Type Number** | **Type Name** | **Description** |
| 10 | Documentation | Comments, messages |
| 20 | Syntax | Spelling, punctuation, typos, instruction formats |
| 30 | Build, Package | Change management, library, version control |
| 40 | Assignment | Declaration, duplicate names, scope, limits |
| 50 | Interface | Procedure calls and references, I/O, user formats |
| 60 | Checking | Error messages, inadequate checks |
| 70 | Data | Structure, content |
| 80 | Function | Logic, pointers, loops, recursion, computation, function defects |
| 90 | System | Configuration, timing, memory |
| 100 | Environment | Design, compile, test, or other support system problems |

Source programs listing

Clase Main:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Program Assignment: 3 \*/

/\* Name: Alvaro Suarez \*/

/\* Date: 9 Feb 2015 \*/

/\* Description: Programa que calcula la regresion lineal, \*/

/\* coeficientes de correlacion y prediccion

/\* mejorada dado un tamaño de proxy \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

import java.io.IOException;

import java.net.URI;

import java.net.URISyntaxException;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.LinkedList;

import java.util.List;

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;

/\*\*

\* es la clase main encargada de ejecutar todos los procedimientos

\*

\*/

public class Main extends HttpServlet

{

//private PathJavaExplorer pje;

@Override

protected void doGet(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException {

showHome(req,resp);

}

private void showHome(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException {

List<CalculadoraDatos> cd = new LinkedList<CalculadoraDatos>();

cd.add(new CalculadoraDatos(1));

cd.add(new CalculadoraDatos(2));

cd.add(new CalculadoraDatos(3));

cd.add(new CalculadoraDatos(4));

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

int aux = i+1;

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

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

resp.getWriter().print("\n Beta0: " + cd.get(i).getBeta0() + " Beta1: "+ cd.get(i).getBeta1());

resp.getWriter().print("\n rxy: " + cd.get(i).getRxy() + " r2: "+ cd.get(i).getR2());

resp.getWriter().print("\n yk: " + cd.get(i).getYk());

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

}

}

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: TablaDatos

import java.util.LinkedList;

import java.util.List;

/\*

\* 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.

\*/

/\*\*

\*

\* @author asan123456

\*/

public class TablaDatos {

private List<TablaDatos> tablaDatos;

private double x;

private double y;

private double x2;

private double xy;

private double y2;

//private int id;

public TablaDatos(double x,double y){

this.x =x;

this.y =y;

}

public TablaDatos(){

super();

}

public void llenarTabla1(){

tablaDatos = new LinkedList<TablaDatos>();

tablaDatos.add(new TablaDatos(130,186));

tablaDatos.add(new TablaDatos(650,699));

tablaDatos.add(new TablaDatos(99,132));

tablaDatos.add(new TablaDatos(150,272));

tablaDatos.add(new TablaDatos(128,291));

tablaDatos.add(new TablaDatos(302,331));

tablaDatos.add(new TablaDatos(95,199));

tablaDatos.add(new TablaDatos(945,1890));

tablaDatos.add(new TablaDatos(368,788));

tablaDatos.add(new TablaDatos(961,1601));

calculadas();

}

//test2

public void llenarTabla2(){

tablaDatos = new LinkedList<TablaDatos>();

tablaDatos.add(new TablaDatos(130.0,15.0));

tablaDatos.add(new TablaDatos(650.0,69.9));

tablaDatos.add(new TablaDatos(99.0,6.5));

tablaDatos.add(new TablaDatos(150.0,22.4));

tablaDatos.add(new TablaDatos(128.0,28.4));

tablaDatos.add(new TablaDatos(302.0,65.9));

tablaDatos.add(new TablaDatos(95.0,19.4));

tablaDatos.add(new TablaDatos(945.0,198.7));

tablaDatos.add(new TablaDatos(368.0,38.8));

tablaDatos.add(new TablaDatos(961.0,138.2));

calculadas();

}

//test3

public void llenarTabla3(){

tablaDatos = new LinkedList<TablaDatos>();

tablaDatos.add(new TablaDatos(163,186));

tablaDatos.add(new TablaDatos(765,699));

tablaDatos.add(new TablaDatos(141,132));

tablaDatos.add(new TablaDatos(166,272));

tablaDatos.add(new TablaDatos(137,291));

tablaDatos.add(new TablaDatos(355,331));

tablaDatos.add(new TablaDatos(136,199));

tablaDatos.add(new TablaDatos(1206,1890));

tablaDatos.add(new TablaDatos(433,788));

tablaDatos.add(new TablaDatos(1130,1601));

calculadas();

}

//test4

public void llenarTabla4(){

tablaDatos = new LinkedList<TablaDatos>();

tablaDatos.add(new TablaDatos(163,15.0));

tablaDatos.add(new TablaDatos(765,69.9));

tablaDatos.add(new TablaDatos(141,6.5));

tablaDatos.add(new TablaDatos(166,22.4));

tablaDatos.add(new TablaDatos(137,28.4));

tablaDatos.add(new TablaDatos(355,65.9));

tablaDatos.add(new TablaDatos(136,19.4));

tablaDatos.add(new TablaDatos(1206,198.7));

tablaDatos.add(new TablaDatos(433,38.8));

tablaDatos.add(new TablaDatos(1130,138.2));

calculadas();

}

private void calculadas(){

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

tablaDatos.get(i).x2 = Math.pow(getTablaDatos().get(i).getX(),2);

tablaDatos.get(i).xy = getTablaDatos().get(i).getX() \* getTablaDatos().get(i).getY();

tablaDatos.get(i).y2 = Math.pow(getTablaDatos().get(i).getY(),2);

}

}

/\*\*

\* @return the tablaDatos

\*/

public List<TablaDatos> getTablaDatos() {

return tablaDatos;

}

/\*\*

\* @return the x

\*/

public double getX() {

return x;

}

/\*\*

\* @return the y

\*/

public double getY() {

return y;

}

/\*\*

\* @return the x2

\*/

public double getX2() {

return x2;

}

/\*\*

\* @return the xy

\*/

public double getXy() {

return xy;

}

/\*\*

\* @return the y2

\*/

public double getY2() {

return y2;

}

}

Clase: CalculadoraDatos

import java.util.LinkedList;

import java.util.List;

/\*

\* 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.

\*/

/\*\*

\*

\* @author asan123456

\*/

public class CalculadoraDatos {

public static final int XK = 386;

private double sumatoriaxi;

private double sumatoriayi;

private double sumatoriax2i;

private double sumatoriaxyi;

private double sumatoriay2i;

private double xavg;

private double yavg;

private double beta0;

private double beta1;

private double rxy;

private double r2;

private double yk;

public CalculadoraDatos(int mostrar){

sumatoriaxi =0;

sumatoriayi =0;

sumatoriax2i =0;

sumatoriaxyi =0;

sumatoriay2i =0;

beta0 =0;

beta1 =0;

rxy =0;

r2 =0;

yk =0;

TablaDatos t= new TablaDatos();

if(mostrar == 1){

t.llenarTabla1();

calcSumatoriasyAVG(t.getTablaDatos());

calcBetas(t.getTablaDatos().size());

calcRs(t.getTablaDatos().size());

calcYk();

}

if(mostrar == 2){

t.llenarTabla2();

calcSumatoriasyAVG(t.getTablaDatos());

calcBetas(t.getTablaDatos().size());

calcRs(t.getTablaDatos().size());

calcYk();

}

if(mostrar == 3){

t.llenarTabla3();

calcSumatoriasyAVG(t.getTablaDatos());

calcBetas(t.getTablaDatos().size());

calcRs(t.getTablaDatos().size());

calcYk();

}

if(mostrar == 4){

t.llenarTabla4();

calcSumatoriasyAVG(t.getTablaDatos());

calcBetas(t.getTablaDatos().size());

calcRs(t.getTablaDatos().size());

calcYk();

}

}

private void calcSumatoriasyAVG(List<TablaDatos> t){

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

sumatoriaxi += t.get(i).getX();

sumatoriayi += t.get(i).getY();

sumatoriax2i += t.get(i).getX2();

sumatoriaxyi += t.get(i).getXy();

sumatoriay2i += t.get(i).getY2();

}

xavg = sumatoriaxi/t.size();

yavg = sumatoriayi/t.size();

}

private void calcBetas(int n){

beta1= (sumatoriaxyi - n\*xavg\*yavg) / (sumatoriax2i - n \* ( Math.pow(xavg, 2) ));

beta0= yavg -getBeta1()\*xavg;

}

private void calcRs(int n){

double denominador1 = n\* sumatoriax2i - Math.pow(sumatoriaxi, 2);

double denominador2 = n\* sumatoriay2i - Math.pow(sumatoriayi, 2);

double denominador = Math.sqrt( denominador1 \* denominador2 );

double numerador = n \* sumatoriaxyi - sumatoriaxi\*sumatoriayi;

rxy= numerador/denominador;

r2 = Math.pow(getRxy(), 2);

}

private void calcYk(){

yk =getBeta0() + getBeta1()\*XK;

}

/\*\*

\* @return the beta0

\*/

public double getBeta0() {

return beta0;

}

/\*\*

\* @return the beta1

\*/

public double getBeta1() {

return beta1;

}

/\*\*

\* @return the rxy

\*/

public double getRxy() {

return rxy;

}

/\*\*

\* @return the r2

\*/

public double getR2() {

return r2;

}

/\*\*

\* @return the yk

\*/

public double getYk() {

return yk;

}

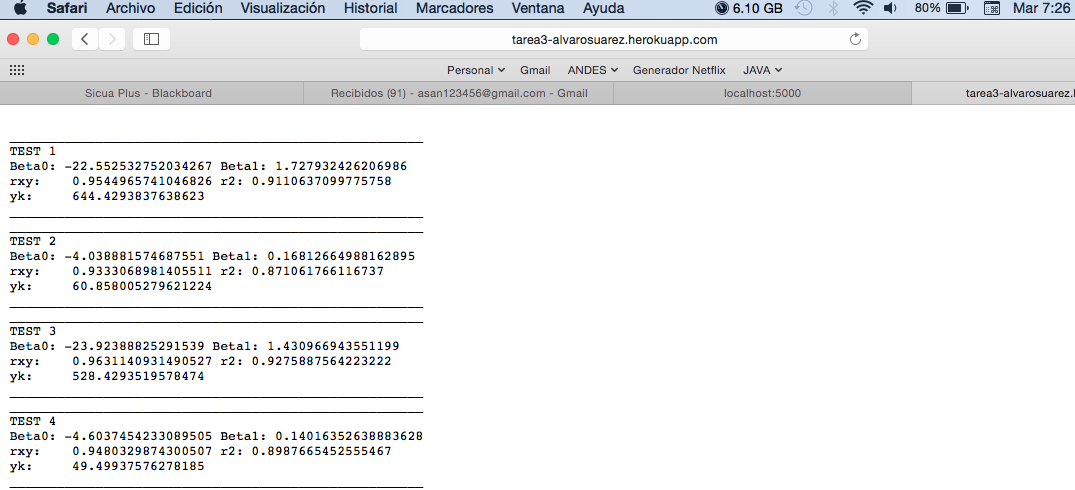
}

Test results

Pruebas Locales:

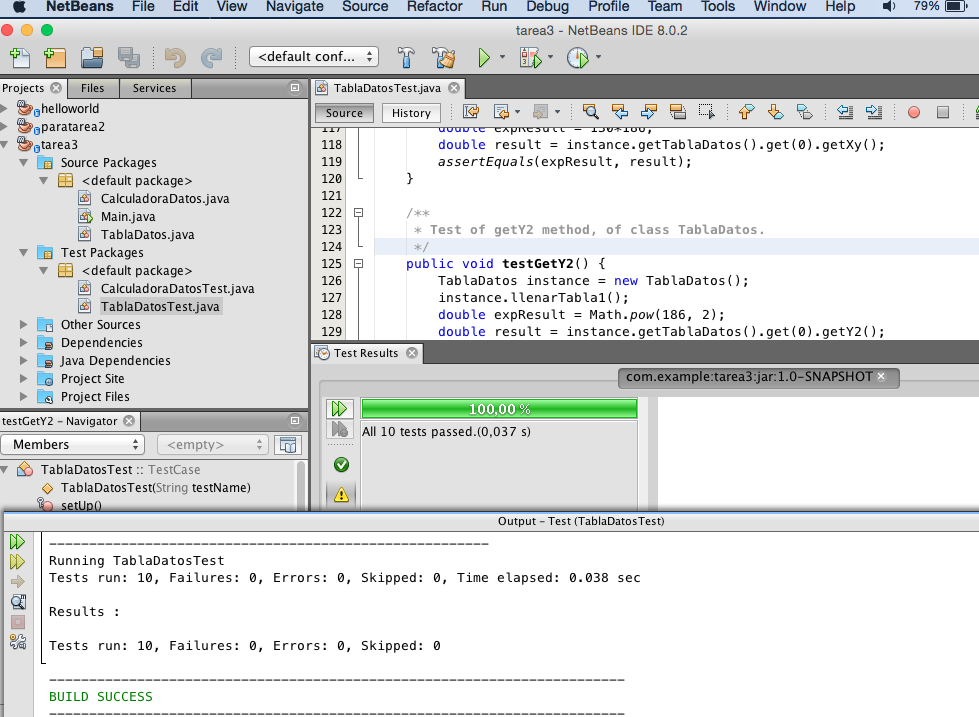


Pruebas en la nube:



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 java.util.List;

import junit.framework.TestCase;

/\*\*

\*

\* @author asan123456

\*/

public class TablaDatosTest extends TestCase {

public TablaDatosTest(String testName) {

super(testName);

}

@Override

protected void setUp() throws Exception {

super.setUp();

}

@Override

protected void tearDown() throws Exception {

super.tearDown();

}

/\*\*

\* Test of llenarTabla1 method, of class TablaDatos.

\*/

public void testLlenarTabla1() {

TablaDatos instance = new TablaDatos();

instance.llenarTabla1();

assertTrue("testLlenarTabla1:fail",instance.getTablaDatos().get(9).getX() == 961);

}

/\*\*

\* Test of llenarTabla2 method, of class TablaDatos.

\*/

public void testLlenarTabla2() {

TablaDatos instance = new TablaDatos();

instance.llenarTabla2();

assertTrue("testLlenarTabla2:fail",instance.getTablaDatos().get(4).getY() == 28.4);

}

/\*\*

\* Test of llenarTabla3 method, of class TablaDatos.

\*/

public void testLlenarTabla3() {

TablaDatos instance = new TablaDatos();

instance.llenarTabla3();

assertTrue("testLlenarTabla3:fail",instance.getTablaDatos().get(1).getX() == 765);

}

/\*\*

\* Test of llenarTabla4 method, of class TablaDatos.

\*/

public void testLlenarTabla4() {

TablaDatos instance = new TablaDatos();

instance.llenarTabla4();

int i = instance.getTablaDatos().size() - 1;

assertTrue("testLlenarTabla4:fail",instance.getTablaDatos().get(i).getY() == 138.2);

}

/\*\*

\* Test of getTablaDatos method, of class TablaDatos.

\*/

public void testGetTablaDatos() {

TablaDatos instance = new TablaDatos();

List<TablaDatos> expResult = instance.getTablaDatos();

List<TablaDatos> result = instance.getTablaDatos();

assertEquals(expResult, result);

}

/\*\*

\* Test of getX method, of class TablaDatos.

\*/

public void testGetX() {

TablaDatos instance = new TablaDatos();

instance.llenarTabla1();

double expResult = 130.0;

double result = instance.getTablaDatos().get(0).getX();

assertEquals(expResult, result);

}

/\*\*

\* Test of getY method, of class TablaDatos.

\*/

public void testGetY() {

TablaDatos instance = new TablaDatos();

instance.llenarTabla1();

double expResult = 186.0;

double result = instance.getTablaDatos().get(0).getY();

assertEquals(expResult, result);

}

/\*\*

\* Test of getX2 method, of class TablaDatos.

\*/

public void testGetX2() {

TablaDatos instance = new TablaDatos();

instance.llenarTabla1();

double expResult = Math.pow(130, 2);

double result = instance.getTablaDatos().get(0).getX2();

assertEquals(expResult, result);

}

/\*\*

\* Test of getXy method, of class TablaDatos.

\*/

public void testGetXy() {

TablaDatos instance = new TablaDatos();

instance.llenarTabla1();

double expResult = 130\*186;

double result = instance.getTablaDatos().get(0).getXy();

assertEquals(expResult, result);

}

/\*\*

\* Test of getY2 method, of class TablaDatos.

\*/

public void testGetY2() {

TablaDatos instance = new TablaDatos();

instance.llenarTabla1();

double expResult = Math.pow(186, 2);

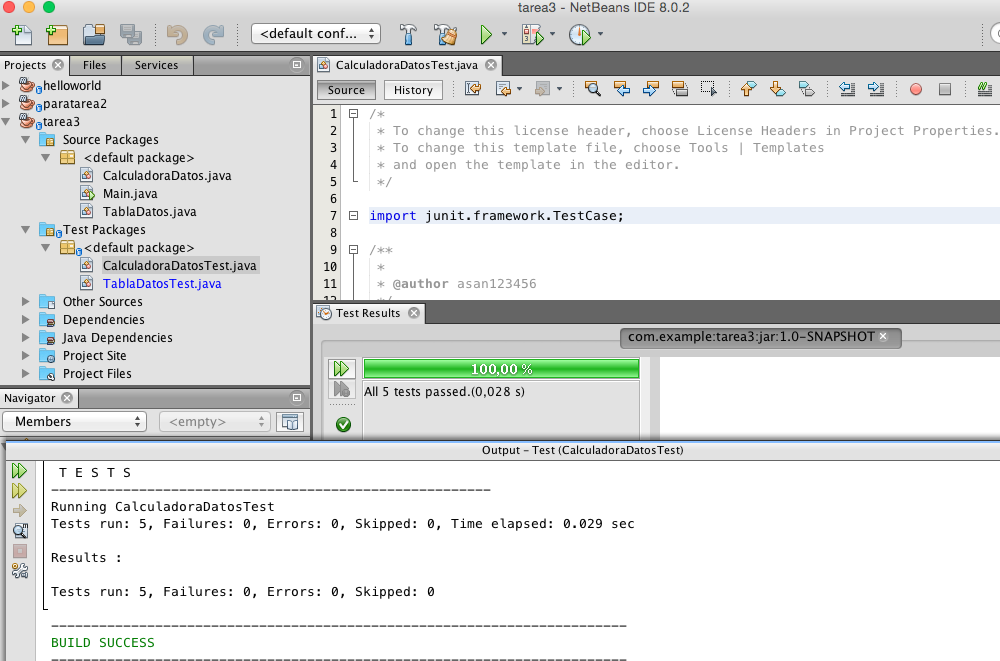
double result = instance.getTablaDatos().get(0).getY2();

assertEquals(expResult, result);

}

}

CalculadorDatosTest



* Clase: CalculadoraDatosTest

/\*

\* 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;

/\*\*

\*

\* @author asan123456

\*/

//-22.55 1.7279 0.9545 0.9111 644.429

public class CalculadoraDatosTest extends TestCase {

public CalculadoraDatosTest(String testName) {

super(testName);

}

@Override

protected void setUp() throws Exception {

super.setUp();

}

@Override

protected void tearDown() throws Exception {

super.tearDown();

}

/\*\*

\* Test of getBeta0 method, of class CalculadoraDatos.

\*/

public void testGetBeta0() {

CalculadoraDatos instance = new CalculadoraDatos(1);

double expResult = -22.55;

double incertidumbre= 0.01;

double result = instance.getBeta0();

assertTrue("testGetBeta0:fail", result + incertidumbre > expResult && result - incertidumbre < expResult);

}

/\*\*

\* Test of getBeta1 method, of class CalculadoraDatos.

\*/

public void testGetBeta1() {

CalculadoraDatos instance = new CalculadoraDatos(1);

double expResult = 1.7279;

double incertidumbre= 0.01;

double result = instance.getBeta1();

//System.out.println("asd: "+instance.getBeta1());

assertTrue("testGetBeta1:fail", result + incertidumbre > expResult && result - incertidumbre < expResult);

}

/\*\*

\* Test of getRxy method, of class CalculadoraDatos.

\*/

public void testGetRxy() {

CalculadoraDatos instance = new CalculadoraDatos(1);

double expResult = 0.9545;

double incertidumbre= 0.01;

double result = instance.getRxy();

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

}

/\*\*

\* Test of getR2 method, of class CalculadoraDatos.

\*/

public void testGetR2() {

CalculadoraDatos instance = new CalculadoraDatos(1);

double expResult = 0.9111;

double incertidumbre= 0.01;

double result = instance.getR2();

assertTrue("testGetR2:fail", result + incertidumbre > expResult && result - incertidumbre < expResult);

}

/\*\*

\* Test of getYk method, of class CalculadoraDatos.

\*/

public void testGetYk() {

CalculadoraDatos instance = new CalculadoraDatos(1);

double expResult = 644.429;

double incertidumbre= 0.01;

double result = instance.getYk();

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

}

}