|  |  |
| --- | --- |
| # tarea4  # git url: https://github.com/axxa/tarea4.git  # heroku url: http://tarea4-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 |

PSP1.1 Project Plan Summary

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Alvaro Andres Suarez Alfonso | Date | 15 Feb 2015 |
| Program | Tarea 4 | Program # | CSOF5101\_01\_4 |
| Instructor | Luis Daniel Benavides Navarro | Language |  |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Summary*** | ***Plan*** | | |  | ***Actual*** | | |  | ***To Date*** | | |
| ***Size/Hour*** | *180/1* | | |  | *230/1* | | |  | *230/1* | | |
| ***Planned Time*** |  | | |  |  | | |  |  | | |
| ***Actual Time*** |  | | |  |  | | |  |  | | |
| ***CPI (Cost-Performance Index)*** | | | |  |  | | |  |  | | |
|  |  | | |  |  | | |  | (Planned/Actual) | | |
| ***% Reused*** |  | | |  |  | | |  |  | | |
| ***% New Reusable*** |  | | |  |  | | |  |  | | |
|  |  | | |  |  | | |  |  | | |
| ***Program Size*** | ***Plan*** | | |  | ***Actual*** | | |  | ***To Date*** | | |
| ***Base (B)*** |  | | |  | 0 | | |  |  | | |
|  |  | | |  | ***(Measured)*** | | |  |  | | |
| ***Deleted (D)*** |  | | |  | 3 | | |  |  | | |
|  |  | | |  | ***(Counted)*** | | |  |  | | |
| ***Modified (M)*** |  | | |  | 5 | | |  |  | | |
|  |  | | |  | ***(Counted)*** | | |  |  | | |
| ***Added (A)*** |  | | |  | 223 | | |  |  | | |
|  |  | | |  | ***(T − B + D − R)*** | | |  |  | | |
| ***Reused (R)*** |  | | |  | 218 | | |  | 841 | | |
|  |  | | |  | ***(Counted)*** | | |  |  | | |
| ***Added and Modified (A+M)*** | 260 | | |  | 228 | | |  | 956 | | |
|  |  | | |  | ***(A + M)*** | | |  |  | | |
| ***Total Size (T)*** |  | | |  | 223 | | |  | 951 | | |
|  |  | | |  | ***(Measured)*** | | |  |  | | |
| ***Total New Reusable*** |  | | |  | 228 | | |  | 841 | | |
|  |  | | |  |  | | |  |  | | |
| **Time in Phase (min.)** | ***Plan*** |  | **Actual** | | |  | **To Date** | | |  | **To Date %** |
| Planning | 10 |  | 5 | | |  | 29 | | |  | 3.71% |
| Design | 10 |  | 7 | | |  | 38 | | |  | 4.87% |
| Code | 80 |  | 69 | | |  | 567 | | |  | 72.69% |
| Compile | 10 |  | 8 | | |  | 49 | | |  | 6.28% |
| Test | 25 |  | 48 | | |  | 85 | | |  | 10.89% |
| Postmortem | 10 |  | 10 | | |  | 112 | | |  | 14.35% |
| Total | 145 |  | 147 | | |  | 780 | | |  | 100% |
|  |  |  |  | | |  |  | | |  |  |
| **Defects Injected** |  |  | **Actual** | | |  | **To Date** | | |  | **To Date %** |
| Planning |  |  | 0 | | |  | 0 | | |  | 0% |
| Design |  |  | 0 | | |  | 0 | | |  | 0% |
| Code |  |  | 1 | | |  | 10 | | |  | 100% |
| Compile |  |  | 0 | | |  | 0 | | |  | 0% |
| Test |  |  | 0 | | |  | 0 | | |  | 0% |
| Total Development |  |  | 1 | | |  | 10 | | |  | 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 |  |  | 1 | | |  | 3 | | |  | 33.33% |
| Total Development |  |  | 1 | | |  | 10 | | |  | 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 | 15 Feb 2015 |
| Program | Tarea 4 | Program # | CSOF5101\_01\_4 |
| Instructor | Luis Daniel Benavides Navarro | Language | Java |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Project** | **Phase** | **Start Date and Time** | **Int. Time** | **Stop Date and Time** | **Delta**  **Time** | **Comments** |
| T4-15\_feb | Plan | 15.56 |  | 16.01 | 5 |  |
|  | Desi | 16.03 |  | 16.10 | 7 |  |
|  | Code | 16.10 |  | 17.19 | 69 |  |
|  | Com | 17.30 |  | 17.38 | 8 |  |
|  | Test | 17.38 |  | 18.26 | 48 |  |
|  | Post | 18.26 |  | 18.36 | 10 |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

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 | 15 Feb 2015 |
| Program | Tarea 4 | Program # | CSOF5101\_01\_4 |
| Instructor | Luis Daniel Benavides Navarro | Language | Java |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
| Tarea 4 |  | | 15 Feb |  | 1 |  | Calculo |  | code |  | test |  | 6 |  |  |
| Description: | | | Calculando average | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| 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: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |

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: 4 \*/

/\* Name: Alvaro Suarez \*/

/\* Date: 14 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 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

{

@Override

protected void doGet(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException {

showHome(req,resp);

}

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

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

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

}

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

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

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

resp.getWriter().print("\n VS: " + cd.getvS() + " S: "+ cd.getS());

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

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

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: ClassInfo

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 ClassInfo {

private List<ClassInfo> tablaDatos;

private String className;

private double LOCMehtod;

private double avg;

private double lnxi;

private double lnxiavg2;

//private int id;

public ClassInfo(String className, double LOCMethod){

this.avg = 0;

this.className =className;

this.LOCMehtod =LOCMethod;

}

public ClassInfo(){

super();

}

public void llenarTabla1(){

tablaDatos = new LinkedList<ClassInfo>();

tablaDatos.add(new ClassInfo("each\_char",6));

tablaDatos.add(new ClassInfo("string\_read",6));

tablaDatos.add(new ClassInfo("single\_character",8.3333));

tablaDatos.add(new ClassInfo("each\_line",10.3333));

tablaDatos.add(new ClassInfo("single\_char",12.3333));

tablaDatos.add(new ClassInfo("string\_builder",16.4000));

tablaDatos.add(new ClassInfo("string\_manager",20.5000));

tablaDatos.add(new ClassInfo("ist\_clump",21.7500));

tablaDatos.add(new ClassInfo("ist\_clip",22.2500));

tablaDatos.add(new ClassInfo("string\_decrementer",23.0000));

tablaDatos.add(new ClassInfo("Char",28.3333));

tablaDatos.add(new ClassInfo("Character",29.0000));

tablaDatos.add(new ClassInfo("Converter",55.8000));

calculadas();

}

public void llenarTabla2(){

tablaDatos = new LinkedList<ClassInfo>();

tablaDatos.add(new ClassInfo("Preface",7));

tablaDatos.add(new ClassInfo("Chapter 1",12));

tablaDatos.add(new ClassInfo("Chapter 2",10));

tablaDatos.add(new ClassInfo("Chapter 3",12));

tablaDatos.add(new ClassInfo("Chapter 4",10));

tablaDatos.add(new ClassInfo("Chapter 5",12));

tablaDatos.add(new ClassInfo("Chapter 6",12));

tablaDatos.add(new ClassInfo("Chapter 7",12));

tablaDatos.add(new ClassInfo("Chapter 8",12));

tablaDatos.add(new ClassInfo("Chapter 9",8));

tablaDatos.add(new ClassInfo("Appendix A",8));

tablaDatos.add(new ClassInfo("Appendix B",8));

tablaDatos.add(new ClassInfo("Appendix C",20));

tablaDatos.add(new ClassInfo("Appendix D",14));

tablaDatos.add(new ClassInfo("Appendix E",18));

tablaDatos.add(new ClassInfo("Appendix F",12));

calculadas();

}

private void calculadas(){

int size = tablaDatos.size();

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

tablaDatos.get(i).lnxi = Math.log(tablaDatos.get(i).LOCMehtod);

this.avg = this.getAvg() + tablaDatos.get(i).lnxi;

}

this.avg = this.getAvg() / size;

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

tablaDatos.get(i).lnxiavg2 = Math.pow(tablaDatos.get(i).lnxi - this.getAvg(),2);

}

}

/\*\*

\* @return the tablaDatos

\*/

public List<ClassInfo> getTablaDatos() {

return tablaDatos;

}

/\*\*

\* @return the LOCMehtod

\*/

public double getLOCMehtod() {

return LOCMehtod;

}

/\*\*

\* @return the lnxi

\*/

public double getLnxi() {

return lnxi;

}

/\*\*

\* @return the lnxiavg2

\*/

public double getLnxiavg2() {

return lnxiavg2;

}

/\*\*

\* @return the avg

\*/

public double getAvg() {

return avg;

}

}

Clase: CalculadoraDatos

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 {

private double stdDev;

private double lnVS;

private double lnS;

private double lnM;

private double lnL;

private double lnVL;

private double vS;

private double s;

private double m;

private double l;

private double vL;

public CalculadoraDatos(int mostrar){

double avg;

ClassInfo t= new ClassInfo();

if(mostrar == 1)

t.llenarTabla1();

if(mostrar == 2)

t.llenarTabla2();

avg = t.getAvg();

stdDev=stdDev(avg,t.getTablaDatos());

lnVS= avg - 2\*stdDev;

lnS= avg - stdDev;

lnM = avg;

lnL = avg + stdDev;

lnVL = avg + 2\*stdDev;

vS =Math.exp(lnVS);

s =Math.exp(lnS);

m =Math.exp(lnM);

l =Math.exp(lnL);

vL =Math.exp(lnVL);

}

public double stdDev(double average,List<ClassInfo> listaDatos){

double result = 0;

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

result = result + listaDatos.get(i).getLnxiavg2();

}

result = result / (listaDatos.size()-1);

return Math.sqrt(result);

}

/\*\*

\* @return the stdDev

\*/

public double getStdDev() {

return stdDev;

}

/\*\*

\* @return the lnVS

\*/

public double getLnVS() {

return lnVS;

}

/\*\*

\* @return the lnS

\*/

public double getLnS() {

return lnS;

}

/\*\*

\* @return the lnM

\*/

public double getLnM() {

return lnM;

}

/\*\*

\* @return the lnL

\*/

public double getLnL() {

return lnL;

}

/\*\*

\* @return the lnVL

\*/

public double getLnVL() {

return lnVL;

}

/\*\*

\* @return the vS

\*/

public double getvS() {

return vS;

}

/\*\*

\* @return the s

\*/

public double getS() {

return s;

}

/\*\*

\* @return the m

\*/

public double getM() {

return m;

}

/\*\*

\* @return the l

\*/

public double getL() {

return l;

}

/\*\*

\* @return the vL

\*/

public double getvL() {

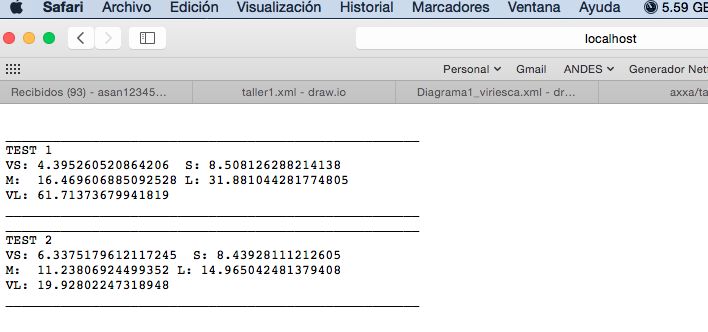
return vL;

}

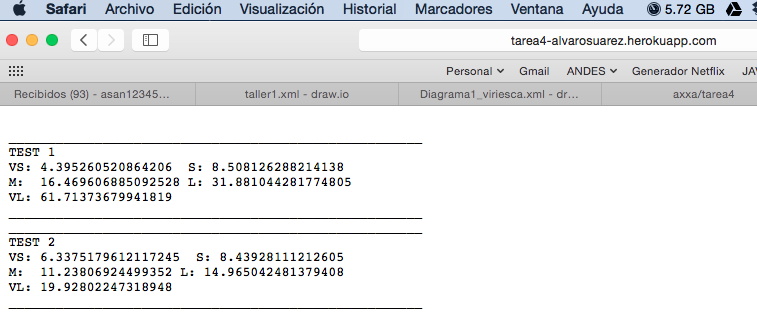
}

Test results

Pruebas Locales:

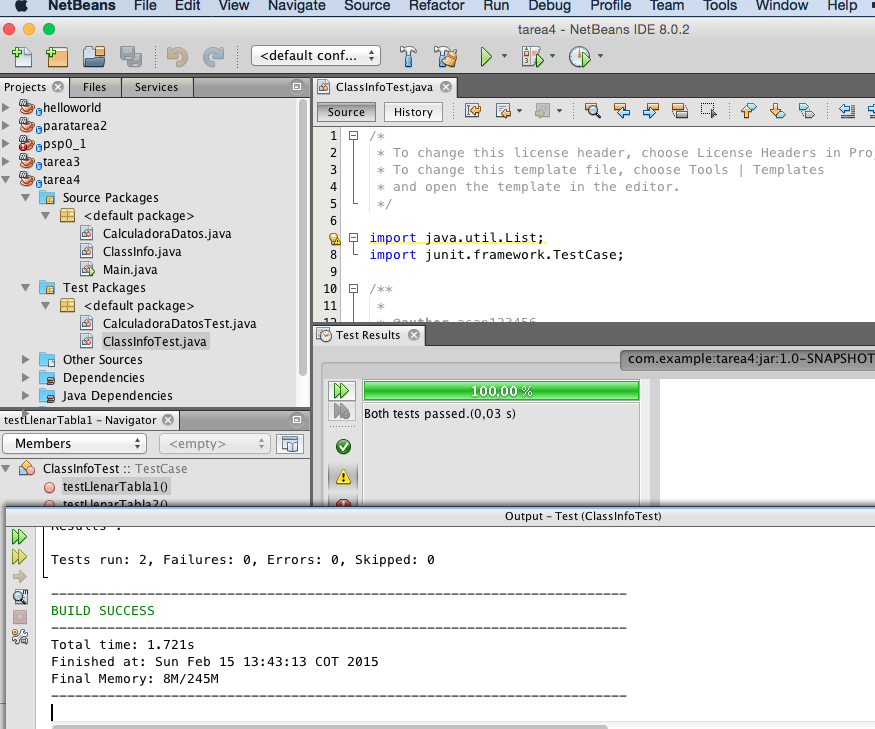


Pruebas en la nube:



Pruebas Unitarias:

TablaDatosTest



* Clase: ClassInfoTest

/\*

\* 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 ClassInfoTest extends TestCase {

/\*\*

\* Test of llenarTabla1 method, of class ClassInfo.

\*/

public void testLlenarTabla1() {

ClassInfo instance = new ClassInfo();

instance.llenarTabla1();

assertTrue("testLlenarTabla1:fail",instance.getTablaDatos().get(0).getLOCMehtod() == 6);

}

/\*\*

\* Test of llenarTabla2 method, of class ClassInfo.

\*/

public void testLlenarTabla2() {

ClassInfo instance = new ClassInfo();

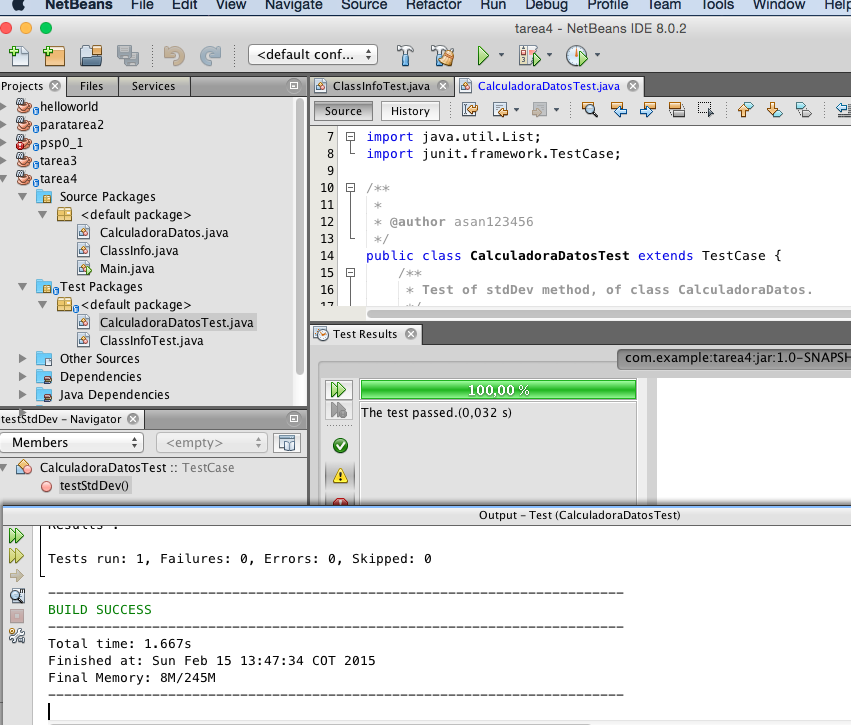
instance.llenarTabla2();

assertTrue("testLlenarTabla2:fail",instance.getTablaDatos().get(0).getLOCMehtod() == 7);

}

}

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 java.util.List;

import junit.framework.TestCase;

/\*\*

\*

\* @author asan123456

\*/

public class CalculadoraDatosTest extends TestCase {

/\*\*

\* Test of stdDev method, of class CalculadoraDatos.

\*/

public void testStdDev() {

CalculadoraDatos instance = new CalculadoraDatos(1);

double average = 2.8015;

double incertidumbre = 0.001;

ClassInfo t= new ClassInfo();

t.llenarTabla1();

List<ClassInfo> listaDatos = t.getTablaDatos();

double expResult = instance.getStdDev();

double result = instance.stdDev(average, listaDatos);

System.out.println("expResult: "+expResult);

System.out.println("result: "+result);

assertTrue("testStdDev:fail",expResult == result);

}

}