|  |
| --- |
| Desarrollo Assignment Kit forProgram 6 |
| Documentos PSP 2.1 |
| Diego Lozano |

TABLA DE CONTENIDO

[1. PSP2.1 Project Plan Summary 3](#_Toc413250847)

[2. PSP Time Recording Log 7](#_Toc413250848)

[3. PSP Defect Recording Log 8](#_Toc413250849)

[4. PSP Defect Type Standard 9](#_Toc413250850)

[5. PSP Process Improvement Proposal (PIP) 10](#_Toc413250851)

[6. Java Coding Standard 11](#_Toc413250852)

[7. Test Report Template 12](#_Toc413250853)

[8. Size Estimating Template 14](#_Toc413250854)

[9. Task Planning Template 16](#_Toc413250855)

[10. Schedule Planning Template 17](#_Toc413250856)

[11. Design Review Checklist 18](#_Toc413250857)

[12. Code Review Checklist 19](#_Toc413250858)

[13. Operational Specification Template 20](#_Toc413250859)

[14. Functional Specification Template 22](#_Toc413250860)

[15. State Specification Template 23](#_Toc413250861)

[16. Logic Specification Template 24](#_Toc413250862)

[17. Metaphor / Architecture Specification Template 25](#_Toc413250863)

[18. Source Program Listing 26](#_Toc413250864)

[19. Test results 33](#_Toc413250865)

1. PSP2.1 Project Plan Summary

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Diego Lozano | Date | 01/03/2015 |
| Program | PSP2.1 | Program # | 6 |
| Instructor | Luis Benavides | Language | Java |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Summary** | | **Plan** | | |  | **Actual** | | |  | **To Date** | | |
| Size/Hour | |  | | |  |  | | |  |  | | |
| Planned Time | |  | | |  |  | | |  |  | | |
| Actual Time | |  | | |  |  | | |  |  | | |
| CPI (Cost-Performance Index) | |  | | |  |  | | |  |  | | |
|  | |  | | |  |  | | |  | (Planned/Actual) | | |
| % Reuse | |  | | |  |  | | |  |  | | |
| % New Reusable | |  | | |  |  | | |  |  | | |
| Test Defects/KLOC or equivalent | |  | | |  |  | | |  |  | | |
| Total Defects/KLOC or equivalent | |  | | |  |  | | |  |  | | |
| Yield % | |  | | |  |  | | |  |  | | |
| ***% Appraisal COQ*** | |  | | |  |  | | |  |  | | |
| ***% Failure COQ*** | |  | | |  |  | | |  |  | | |
| ***COQ A/F Ratio*** | |  | | |  |  | | |  |  | | |
| ***PQI*** | |  | | |  |  | | |  |  | | |
|  | |  | | |  |  | | |  |  | | |
| **Program Size** | | **Plan** | | |  | **Actual** | | |  | **To Date** | | |
| Base (B) | |  | | |  |  | | |  |  | | |
|  | | (Measured) | | |  | (Measured) | | |  |  | | |
| Deleted (D) | |  | | |  |  | | |  |  | | |
|  | | (Estimated) | | |  | (Counted) | | |  |  | | |
| Modified (M) | |  | | |  |  | | |  |  | | |
|  | | (Estimated) | | |  | (Counted) | | |  |  | | |
| Added (A) |  | | | |  |  | | |  |  | | |
|  | (A+M − M) | | | |  | (T − B + D − R) | | |  |  | | |
| Reused (R) |  | | | |  |  | | |  |  | | |
|  | (Estimated) | | | |  | (Counted) | | |  |  | | |
| Added and Modified (A+M) |  | | | |  |  | | |  |  | | |
|  | (Projected) | | | |  | (A + M) | | |  |  | | |
| Total Size (T) |  | | | |  |  | | |  |  | | |
|  | (A+M + B − M − D + R) | | | |  | (Measured) | | |  |  | | |
| Total New Reusable |  | | | |  |  | | |  |  | | |
|  |  | | | |  |  | | |  |  | | |
| Estimated Proxy Size (E) |  | | | |  |  | | |  |  | | |
|  | |  | | |  |  | | |  |  | | |
| ***Upper Prediction Interval (70%)*** | |  | | |  |  | | |  |  | | |
| ***Lower Prediction Interval (70%)*** | |  | | |  |  | | |  |  | | |
|  | |  |  |  | | |  |  | | |  |  |

**(continued)**

**PSP2.1 Project Plan Summary (continued)**

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Diego Lozano | Program # | 6 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Time in Phase (min.)** | **Plan** |  | | **Actual** | | | | |  | | **To Date** | | |  | | **To Date %** | |
| Planning |  |  | |  | | | | |  | |  | | |  | |  | |
| Design |  |  | |  | | | | |  | |  | | |  | |  | |
| Design Review |  |  | |  | | | | |  | |  | | |  | |  | |
| Code |  |  | |  | | | | |  | |  | | |  | |  | |
| Code Review |  |  | |  | | | | |  | |  | | |  | |  | |
| Compile |  |  | |  | | | | |  | |  | | |  | |  | |
| Test |  |  | |  | | | | |  | |  | | |  | |  | |
| Postmortem |  |  | |  | | | | |  | |  | | |  | |  | |
| Total |  |  | |  | | | | |  | |  | | |  | |  | |
| ***Total Time UPI (70%)*** |  |  | |  | | | | |  | |  | | |  | |  | |
| ***Total Time LPI (70%)*** |  |  | |  | | | | |  | |  | | |  | |  | |
|  |  | |  | |  | | |  | |  | | | | |  | |  |
| **Defects Injected** | **Plan** | |  | | **Actual** | | |  | | **To Date** | | | | |  | | **To Date %** |
| Planning |  | |  | |  | | |  | |  | | | | |  | |  |
| Design |  | |  | |  | | |  | |  | | | | |  | |  |
| Design Review |  | |  | |  | | |  | |  | | | | |  | |  |
| Code |  | |  | |  | | |  | |  | | | | |  | |  |
| Code Review |  | |  | |  | | |  | |  | | | | |  | |  |
| Compile |  | |  | |  | | |  | |  | | | | |  | |  |
| Test |  | |  | |  | | |  | |  | | | | |  | |  |
| Total Development |  | |  | |  | | |  | |  | | | | |  | |  |
|  |  | |  | |  | | |  | |  | | | | |  | |  |
| **Defects Removed** | **Plan** | |  | | **Actual** | | |  | | **To Date** | | | | |  | | **To Date %** |
| Planning |  | |  | |  | | |  | |  | | | | |  | |  |
| Design |  | |  | |  | | |  | |  | | | | |  | |  |
| Design Review |  | |  | |  | | |  | |  | | | | |  | |  |
| Code |  | |  | |  | | |  | |  | | | | |  | |  |
| Code Review |  | |  | |  | | |  | |  | | | | |  | |  |
| Compile |  | |  | |  | | |  | |  | | | | |  | |  |
| Test |  | |  | |  | | |  | |  | | | | |  | |  |
| Total Development |  | |  | |  | | |  | |  | | | | |  | |  |
| After Development |  | |  | |  | | |  | |  | | | | |  | |  |
|  |  | | | | |  |  | | | | |  |  | | | | |
| **Defect Removal Efficiency** | **Plan** | | | | |  | **Actual** | | | | |  | **To Date** | | | | |
| Defects/Hour − Design Review |  | | | | |  |  | | | | |  |  | | | | |
| Defects/Hour − Code Review |  | | | | |  |  | | | | |  |  | | | | |
| Defects/Hour − Compile |  | | | | |  |  | | | | |  |  | | | | |
| Defects/Hour − Test |  | | | | |  |  | | | | |  |  | | | | |
| DRL (DLDR/UT) |  | | | | |  |  | | | | |  |  | | | | |
| DRL (Code Review/UT) |  | | | | |  |  | | | | |  |  | | | | |
| DRL (Compile/UT) |  | | | | |  |  | | | | |  |  | | | | |

|  |  |  |
| --- | --- | --- |
| PSP2.1 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. * A part could be a module, component, product, or system. | |
| 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. | |
| Summary | * Enter the added and modified size per hour planned, actual, and to-date. * Enter the planned and actual times for this program and prior programs. * For planned time to date, use the sum of the current planned time and the to-date planned time for the most recent prior program. * CPI = (To Date Planned Time)/(To Date Actual Time). * Reuse % is reused size as a percentage of total program size. * New Reusable % is new reusable size as a percentage of added and modified size. * Enter the test and total defects/KLOC or other appropriate measure. * Enter the planned, actual, and to-date yield before compile. | |
| *Quality Indicators* | * ***Appraisal COQ: the percentage of development time in reviews.*** * ***Failure COQ: the percentage of development time in compile and test.*** * ***A/FR: the ratio of appraisal to failure COQ.*** * ***Enter the planned, actual, and to-date PQI (the process quality index)*** | |
| Program Size | * Enter plan base, deleted, modified, reused, new reusable, and total size from the Size Estimating template. * Enter the plan added and modified size value (A+M) from projected added and modified size (P) on the Size Estimating template. * Calculate plan added size as A+M – M. * Enter estimated proxy size (E) from the Size Estimating template. * Enter actual base, deleted, modified, reused, total, and new reusable size from the Size Estimating template. * 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 plan total time in phase from the estimated total development time on the Size Estimating template. * 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. | |
| *Prediction Interval* | * ***Enter the 70% UPI and LPI total size and time ranges.*** | |

**(continued)**

|  |  |  |
| --- | --- | --- |
| PSP2.1 Plan Summary Instructions (continued) | |  |
| Defects Injected | * Enter the total estimated defects injected. * Distribute the estimated total defects across the development phases according to the To Date % for the most recently developed program. * 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 | * Enter the estimated total defects removed. * Distribute the estimated total defects across the development phases according to the To Date % for the most recently developed program. * 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. | |
| Defect-Removal Efficiency | * Calculate and enter the defects removed per hour in design review, code review, compile, and test. * For DRL, take the ratio of the review and compile rates with test. * Where there were no test defects, use the to-date test defect/hour value. | |

1. PSP Time Recording Log

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Diego Lozano | Date | 01/03/2015 |
| Program | PSP2.1 | Program # | 6 |
| Instructor | Luis Benavides | Language | Java |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Project** | **Phase** | **Start Date and Time** | **Int. Time** | **Stop Date and Time** | **Delta**  **Time** | **Comments** |
| PSP2.1 | Planning | 01/03/2015 08:30 |  | 01/03/2015 08:45 | 15 | Produce or obtain a requirements statement |
| PSP2.1 | Planning | 01/03/2015 08:45 |  | 01/03/2015 9:05 | 20 | Use the PROBE method to estimate the added and modified size and the size prediction interval of this program. |
| PSP2.1 | Planning | 01/03/2015 9:05 |  | 01/03/2015 9:30 | 25 | Complete the Size Estimating template. |
| PSP2.1 | Planning | 01/03/2015 10:00 |  | 01/03/2015 10:30 | 30 | Use the PROBE method to estimate the required development time and the time prediction interval. |
| PSP2.1 | Planning | 01/03/2015 10:45 |  | 01/03/2015 11:00 | 15 | Enter the plan data in the Project Plan Summary form. |
| PSP2.1 | Planning | 01/03/2015 11:00 |  | 01/03/2015 11:20 | 20 | Complete the Time Recording log. |
| PSP2.1 | Development | 01/03/2015 11:35 |  | 01/03/2015 11:50 | 15 | Design the program |
| PSP2.1 | Development | 01/03/2015 11:50 |  | 01/03/2015 12:00 | 10 | Review the design |
| PSP2.1 | Development | 01/03/2015 12:00 | 30 | 01/03/2015 13:30 | 60 | Implement the design |
| PSP2.1 | Development | 01/03/2015 13:30 |  | 01/03/2015 13:50 | 20 | Review the code |
| PSP2.1 | Development | 01/03/2015 13:50 |  | 01/03/2015 14:00 | 10 | Compile the program |
| PSP2.1 | Development | 01/03/2015 14:00 | 15 | 01/03/2015 15:00 | 45 | Test the program, and fix and log all defects found |
| PSP2.1 | Development | 01/03/2015 15:00 |  | 01/03/2015 15:35 | 35 | Complete the Time Recording log |
| PSP2.1 | Postmortem | 01/03/2015 15:35 |  | 01/03/2015 16:20 | 45 | Complete the Project Plan Summary form with actual time, defect, and size data |
| PSP2.1 | Postmortem | 01/03/2015 16:20 |  | 01/03/2015 16:25 | 5 | Complete the Time Recording log |

1. 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 | Diego Lozano | Date | 01/03/2015 |
| Program | PSP2.1 | Program # | 6 |
| Instructor | Luis Benavides | Language | Java |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
| PSP 2.1 |  | | 01/03/2015 |  | 1 |  | 80 |  | Design |  | Dev |  | 5 |  | X |
| Description: | | | Error en la lógica del proceso, los datos no dan, por problemas con los cálculos. | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| Project |  | | Date |  | Number |  | Type |  | Inject |  | Remove |  | Fix Time |  | Fix Ref. |
| PSP 2.1 |  | | 01/03/2015 |  | 2 |  | 80 |  | Design |  | Dev |  | 20 |  | X |
| Description: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| 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: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
| 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: | | |  | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |

1. 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 |

1. PSP Process Improvement Proposal (PIP)

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Diego Lozano | Date | 01/03/2015 |
| Program | PSP2.1 | Program # | 6 |
| Instructor | Luis Benavides | Language | Java |

|  |
| --- |
| Problem Description |
| Briefly describe the problems that you encountered. |
|  |
| Los problemas que encontré fueron de interpretación de los requerimientos, ya que se encontraban escritos |
| de manera ambigua impactando en el momento del desarrollo en dos errores. |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| Proposal Description |
| Briefly describe the process improvements that you propose. |
|  |
| Mejorar los procesos de descripción de los problemas, y si se detalla un proceso, hacerlo de una manera |
| más sencilla. |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| Other Notes and Comments |
| Note any other comments or observations that describe your experiences or improvement ideas. |
|  |
| Se debería realizar una validación de los diseños antes de realizar un desarrollo y mejorar los estimados de |
| los tiempos. |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

|  |  |
| --- | --- |
| 1. Java Coding Standard | |
| Purpose | To guide implementation of Java programs |
| Program Headers | Begin all programs with a descriptive header. |
| Header Format | /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\* Program Assignment: the program number \*/  /\* Name: your name \*/  /\* Date: the date you started developing the program \*/  /\* Description: a short description of the program and what it does \*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ |
| Identifiers | Use descriptive names for all variable, function names, constants, and other identifiers. Avoid abbreviations or single-letter variables. |
| Identifier Example | Int number\_of\_students; /\* This is GOOD \*/  Float: x4, j, ftave; /\* This is BAD \*/ |
| Comments | Document the code so the reader can understand its operation.  Comments should explain both the purpose and behavior of the code.  Comment variable declarations to indicate their purpose. |
| Good Comment | If(record\_count > limit) /\* have all records been processed? \*/ |
| Bad Comment | If(record\_count > limit) /\* check if record count exceeds limit \*/ |
| Major Sections | Precede major program sections by a block comment that describes the processing done in the next section. |
| Example | /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\* The program section examines the contents of the array ‘grades’ and calcu- \*/  /\* lates the average class grade. \*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ |
| Blank Spaces | Write programs with sufficient spacing so they do not appear crowded.  Separate every program construct with at least one space. |
| Indenting | Indent each brace level from the preceding level.  Open and close braces should be on lines by themselves and aligned. |
| Indenting Example | while (miss\_distance > threshold)  {  success\_code = move\_robot (target \_location);  if (success\_code == MOVE\_FAILED)  {  printf(“The robot move has failed.\n”);  }  } |
| Capitalization | Capitalize all defines.  Lowercase all other identifiers and reserved words.  To make them readable, user messages may use mixed case. |
| Capitalization Examples | #define DEFAULT-NUMBER-OF-STUDENTS 15  int class-size = DEFAULT-NUMBER-OF-STUDENTS; |

1. Test Report Template

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Diego Lozano | Date | 01/03/2015 |
| Program | PSP2.1 | Program # | 6 |
| Instructor | Luis Benavides | Language | Java |

|  |  |
| --- | --- |
| Test Name/Number | T-PSP2.1-01 |
| Test Objective | Realizar una prueba parcial del desarrollo del programa PSP2.1, con el fin de |
|  | determinar posibles errores en la lógica. |
| Test Description | Se necesita realizar el test sobre los datos de entrada de ejemplo, y mínimo |
|  | debe de ejecutar la aplicación correctamente . |
|  |  |
|  |  |
|  |  |
| Test Conditions | Los datos de entada deben corresponder a los datos de ejemplo. |
|  |  |
|  |  |
|  |  |
|  |  |
| Expected Results | Los datos generados deben corresponder a la hoja de resultados de la tarea. |
|  |  |
|  |  |
|  |  |
| Actual Results | El resultado fue fallido, a pesar de que se generó sin errores, los datos no |
|  | coincidieron con los esperados. |
|  |  |
|  |  |
|  |  |
| Test Name/Number | T-PSP2.1-02 |
| Test Objective | Realizar una prueba parcial del desarrollo del programa PSP2.1 después de |
|  | realizar unos ajustes respecto al caso anterior, con el fin de determinar posibles errores en la lógica. |
| Test Description | Se necesita realizar el test sobre los datos de entrada de ejemplo, y mínimo |
|  | debe de ejecutar la aplicación correctamente. |
|  |  |
|  |  |
|  |  |
| Test Conditions | Los datos de entada deben corresponder a los datos de ejemplo. |
|  |  |
|  |  |
|  |  |
|  |  |
| Expected Results | Los datos generados deben corresponder a la hoja de resultados de la tarea, |
|  | y la aplicación debe funcionar correctamente. |
|  |  |
|  |  |
| Actual Results | El resultado fue fallido, se generó con errores, los datos no coinciden con los |
|  | esperados. |
|  |  |
|  |  |
|  |  |
| Test Name/Number | T-PSP2.1-03 |
| Test Objective | Realizar una prueba parcial del desarrollo del programa PSP2.1 después de |
|  | realizar unos ajustes respecto al caso anterior, con el fin de determinar posibles errores en la lógica. |
| Test Description | Se necesita realizar el test sobre los datos de entrada de ejemplo, y mínimo |
|  | debe de ejecutar la aplicación correctamente |
|  |  |
|  |  |
|  |  |
| Test Conditions | Los datos de entada deben corresponder a los datos de ejemplo. |
|  |  |
|  |  |
|  |  |
|  |  |
| Expected Results | Los datos generados deben corresponder a la hoja de resultados de la tarea. |
|  |  |
|  |  |
|  |  |
| Actual Results | El resultado fue exitoso, con algunos errores en diseño gráfico. |
|  |  |
|  |  |
|  |  |
|  |  |
| Test Name/Number | T-PSP2.1-04 |
| Test Objective | Realizar una prueba total del desarrollo del programa PSP2.1. |
|  |  |
| Test Description | Se necesita realizar el test sobre los datos de entrada de ejemplo y todos los |
|  | ejercicios propuestos, y debe ejecutarla exitosamente. |
|  |  |
|  |  |
|  |  |
| Test Conditions | Los datos de entada deben ser los del ejemplo y los propuestos. |
|  |  |
|  |  |
|  |  |
|  |  |
| Expected Results | Los datos generados deben corresponder a la hoja de resultados de la tarea, |
|  | y la aplicación debe funcionar correctamente. |
|  |  |
|  |  |
| Actual Results | Los datos generados corresponden a los mostrados en la hoja de la tarea. Y |
|  | los demás datos son consistentes. |
|  |  |
|  |  |
|  |  |

1. Size Estimating Template

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Diego Lozano | Date | 01/03/2015 |
| Program | PSP2.1 | Program # | 6 |
| Instructor | Luis Benavides | Language | Java |
| Size Measure | 350 |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Estimated | | | | | | | | | | | |
| Base Parts |  | Base | |  | Deleted | |  | Modified | | |  | Added | |
|  |  |  | |  |  | |  |  | | |  |  | |
| Impresión pantalla |  | 60 | |  | 15 | |  | 20 | | |  | 18 | |
| Controlador |  | 35 | |  | 10 | |  | 33 | | |  | 26 | |
|  |  |  | |  |  | |  |  | | |  |  | |
| Total | B | | 95 | D | | 25 | M | | 53 | **BA** | | | 44 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Actual | | | | | | | | | | |
| Base Parts |  | Base | |  | Deleted | |  | Modified | |  | Added | |
|  |  |  | |  |  | |  |  | |  |  | |
| Impresión pantalla |  | 75 | |  | 37 | |  | 26 | |  | 24 | |
| Controlador |  | 40 | |  | 13 | |  | 5 | |  | 15 | |
|  |  |  | |  |  | |  |  | |  |  | |
| Total |  | | 115 |  | | 50 |  | | 31 |  | | 39 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Estimated | | | | | | | |  | Actual | | |
| Parts Additions |  | Type |  | Items |  | Rel. Size | |  | Size\* |  | Size\* |  | Items |
|  |  |  |  |  |  |  | |  |  |  |  |  |  |
| Cálculos |  | Math |  | 6 |  | 150 | |  | 80 |  | 100 |  | 5 |
|  |  |  |  |  |  |  | |  |  |  |  |  |  |
|  |  |  |  |  |  |  | |  |  |  |  |  |  |
|  |  |  |  |  |  |  | |  |  |  |  |  |  |
|  |  |  |  |  |  |  | |  |  |  |  |  |  |
|  |  |  |  |  |  |  | |  |  |  |  |  |  |
|  |  |  |  |  |  |  | |  |  |  |  |  |  |
|  |  |  |  |  |  |  | |  |  |  |  |  |  |
|  |  |  |  |  |  |  | |  |  |  |  |  |  |
|  |  |  |  |  |  |  | |  |  |  |  |  |  |
| Total |  |  |  |  |  |  | PA | | 80 |  | 100 |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | |  | Estimated |  | Actual |
| Reused Parts | |  | Size |  | Size |
|  | |  |  |  |  |
| Impresión pantalla | |  | 20 |  | 12 |
|  | |  |  |  |  |
|  | |  |  |  |  |
|  | |  |  |  |  |
|  | |  |  |  |  |
|  | |  |  |  |  |
|  | |  |  |  |  |
| Total | R | | 20 |  | 12 |

**(continued)**

Size Estimating Template (continued)

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Diego Lozano | Program | PSP2.1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PROBE Calculation Worksheet (Added and Modified) |  | Size |  | Time |
| Added size (A): A = BA+PA |  | 124 |  |  |
| Estimated Proxy Size (E): E = BA+PA+M |  | 177 |  |  |
| PROBE estimating basis used: (A, B, C, or D) |  | 214 |  | 250 |
| Correlation: (R2) |  |  |  |  |
| Regression Parameters: β0 Size and Time |  |  |  |  |
| Regression Parameters: β1 Size and Time |  |  |  |  |
| Projected Added and Modified Size (P): P = β0size + β1size\*E |  |  |  |  |
| Estimated Total Size (T): T = P + B - D - M + R |  | 350 |  |  |
| Estimated Total New Reusable (NR): sum of \* items |  | 1 |  |  |
| Estimated Total Development Time: Time = β0time + β1time\*E |  |  |  |  |
| Prediction Range: Range |  |  |  |  |
| Upper Prediction Interval: UPI = P + Range |  |  |  |  |
| Lower Prediction Interval: LPI = P - Range |  |  |  |  |
| Prediction Interval Percent: |  |  |  |  |

1. Task Planning Template

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Diego Lozano | Date | 01/03/2015 |
| Program | PSP2.1 | Program # | 6 |
| Instructor | Luis Benavides | Language | Java |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Task | | | Plan | | | | | | | Actual | | |
| Program/Part | Phase | Task Name | Task Hours | Cumulative Task Hours | Week Due | Week | Week Predicted | Planned Value (PV) | Cumulative  PV | Task Hours | Cumulative EV | Week |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| Totals | | |  |  |  |  |  |  |  |  |  |  |

1. Schedule Planning Template

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Diego Lozano | Date | 01/03/2015 |
| Program | PSP2.1 | Program # | 6 |
| Instructor | Luis Benavides | Language | Java |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Plan | | | Actual | | | | Predicted |
| Week  No. | Date | Schedule  Hours | Cumulative Schedule Hours | Cumulative  Planned  Value | Schedule  Hours | Cumulative  Schedule Hours | Week  Earned Value | Cumulative  Earned Value | Cumulative  Predicted Earned Value |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

1. Design Review Checklist

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Diego Lozano | Date | 01/03/2015 |
| Program | PSP2.1 | Program # | 6 |
| Instructor | Luis Benavides | Language | JAVA |

|  |  |
| --- | --- |
| Purpose | To guide you in conducting an effective design review |
| General | Review the entire program for each checklist category; do not attempt to review for more than one category at a time!  As you complete each review step, check off that item in the box at the right.  Complete the checklist for one program or program unit before reviewing the next. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | YES | NO | N/A |
| Complete | Verify that the design covers all of the applicable requirements.  All specified outputs are produced. | X |  |  |
| Logic | Verify that program sequencing is proper.  Stacks, lists, and so on are in the proper order.  Recursion unwinds properly.  Verify that all loops are properly initiated, incremented, and terminated.  Examine each conditional statement and verify all cases. | X |  |  |
| Special Cases | Check all special cases.  Ensure proper operation with empty, full, minimum, maximum, negative, and zero values for all variables.  Protect against out-of-limits, overflow, and underflow conditions.  Ensure “impossible” conditions are absolutely impossible.  Handle all possible incorrect or error conditions. | X |  |  |
| Functional Use | Verify that all functions, procedures, or methods are fully understood and properly used.  Verify that all externally referenced abstractions are precisely defined. | X |  |  |
| System Considerations | Verify that the program does not cause system limits to be exceeded.  Verify that all safety conditions conform to the safety specifications. | X |  |  |
| Names | Verify that  all special names are clear, defined, and authenticated  the scopes of all variables and parameters are self-evident or defined  all named items are used within their declared scopes | X |  |  |

1. Code Review Checklist

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Diego Lozano | Date | 01/03/2015 |
| Program | PSP2.1 | Program # | 6 |
| Instructor | Luis Benavides | Language | Java |

|  |  |
| --- | --- |
| Purpose | To guide you in conducting an effective code review |
| General | Review the entire program for each checklist category; do not attempt to review for more than one category at a time!  As you complete each review step, check off that item in the box at the right.  Complete the checklist for one program or program unit before reviewing the next. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | YES | NO | N/A |
| Complete | Verify that the code covers all of the design. | X |  |  |
| Initialization | Check variable and parameter initialization.  at program initiation  at start of every loop  at class/function/procedure entry | X |  |  |
| Names | Check name spelling and use.  Is it consistent?  Is it within the declared scope?  Do all structures and classes use ‘.’ reference? | X |  |  |
| Pointers | Check that all  pointers are initialized NULL  pointers are deleted only after new | X |  |  |
| () Pairs | Ensure that () are proper and matched. | X |  |  |
| Logic Operators | Verify the proper use of ==, =, ||, and so on.  Check every logic function for (). | X |  |  |
| Standards | Ensure that the code conforms to the coding standards. | X |  |  |
| File Open and Close | Verify that all files are  properly declared  opened  closed | X |  |  |
| Class Definition | - Does each class have an appropriate constructor?  - Can the class inheritance hierarchy be simplified? | X |  |  |
| Computation/Numeric | - Are parentheses used to avoid ambiguity?  - Are divisors tested for zero or noise? | X |  |  |
| Control Flow | - Will all loops terminate?  - Does each switch statement have a default case?  - Does every method terminate?  - Are all exceptions handled appropriately? | X |  |  |
| Comment | - Does every method, class, and file have an appropriate header comment?  - Does every attribute,variable or constant declaration have a comment? | X |  |  |
| Performance | - Can better data structures or more efficient algorithms be used?  - Are timeouts or error traps used for external device accesses? | X |  |  |

1. Operational Specification Template

|  |  |  |  |
| --- | --- | --- | --- |
| **Student** | Diego Lozano | **Date** | 01/03/2015 |
| **Program** | PSP2.1 | **Program #** | 6 |
| **Instructor** | Luis Benavides | **Language** | Java |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Scenario Number** | **1** | **User Objective** | Realizar los cálculos de las integrales | |
| **Scenario Objective** | |  | | |
| **Source** | **Step** | **Action** | | **Comments** |
| Usuario | 1 | Ingresar a la pagina | |  |
| Sistema | 2 | Muestra la página inicial | |  |
| Usuario | 3 | Hace clic en el botón calcular | |  |
| Sistema | 4 | Muestra la información calculada del proceso | | Realiza las integrales con los valores establecidos |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |
|  |  |  | |  |

**Operational Specification Template Instructions**

|  |  |
| --- | --- |
| Purpose | * To hold descriptions of the likely operational scenarios followed during program use * To ensure that all significant usage issues are considered during program design * To specify test scenarios |
| General | * Use this template for complete programs, subsystems, or systems. * Group multiple small scenarios on a single template, as long as they are clearly distinguished and have related objectives. * List the major scenarios and reference other exception, error, or special cases under comments. * Use this template to document the operational specifications during planning, design, test development, implementation, and test. * After implementation and testing, update the template to reflect the actual implemented product. |
| 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. |
| Scenario Number | Where several scenarios are involved, reference numbers are needed. |
| User Objective | List the users’ likely purpose for the scenario, for example, to log onto the system or to handle an error condition. |
| Scenario Objective | List the designer’s purpose for the scenario, for example, to define common user errors or to detail a test scenario. |
| Source | * Enter the source of the scenario action. * Example sources could be user, program, and system. |
| Step | Provide sequence numbers for the scenario steps. These facilitate reviews and inspections. |
| Action | Describe the action taken, such as   * Enter incorrect mode selection. * Provide error message. |
| Comments | List significant information relating to the action, such as   * User enters an incorrect value. * An error is possible with this action. |

1. Functional Specification Template

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Student** | | | Diego Lozano | | **Date** | 01/03/2015 |
| **Program** | | | PSP2.1 | | **Program #** | 6 |
| **Instructor** | | | Luis Benavides | | **Language** | Java |
|  | | | | | | |
| **Class Name** | |  | | | | |
| **Parent Class** | |  | | | | |
|  | |  | | | | |
|  | |  | | | | |
|  | |  | | | | |
|  | | | | | | |
| **Attributes** | | | | | | |
|  | **Declaration** | | | **Description** | | |
|  |  | | |  | | |
|  |  | | |  | | |
|  |  | | |  | | |
|  |  | | |  | | |
|  |  | | |  | | |
|  |  | | |  | | |
|  | | | | | | |
| **Items** | | | | | | |
|  | **Declaration** | | | **Description** | | |
|  |  | | |  | | |
|  |  | | |  | | |
|  |  | | |  | | |
|  |  | | |  | | |
|  |  | | |  | | |
|  |  | | |  | | |
|  |  | | |  | | |
|  |  | | |  | | |
|  |  | | |  | | |
|  |  | | |  | | |

1. State Specification Template

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Student | | Diego Lozano | | | Date | | 01/03/2015 |
| Program | | PSP2.1 | | | Program # | | 6 |
| Instructor | | Luis Benavides | | | Language | | Java |
|  | | |  | | | | |
| **State Name** | | | | **Description** | | | |
|  | | | |  | | | |
|  | | | |  | | | |
|  | | | |  | | | |
|  | | | |  | | | |
|  | | | |  | | | |
| **Function/Parameter** | | | | **Description** | | | |
|  | | | |  | | | |
|  | | | |  | | | |
|  | | | |  | | | |
|  | | | |  | | | |
|  | | | |  | | | |
|  | | | |  | | | |
| **States/Next States** | | | | **Transition Condition** | | **Action** | |
|  | | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |
|  | | |  | |  | |

1. Logic Specification Template

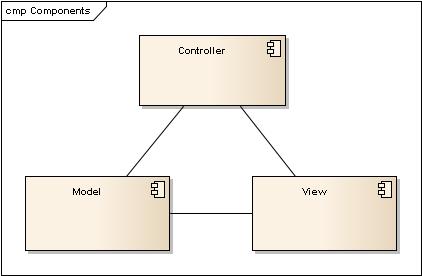
|  |  |  |  |
| --- | --- | --- | --- |
| Student | Diego Lozano | Date | 01/03/2015 |
| Program | PSP2.1 | Program # | 6 |
| Instructor | Luis Benavides | Language | Java |

|  |  |
| --- | --- |
| **Design** |  |
| **References** |  |
|  |  |
|  |  |
|  |  |

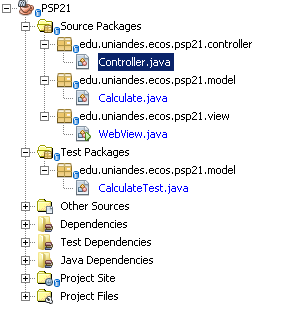
|  |  |
| --- | --- |
| **Parameters** |  |
|  |  |
|  |  |
|  |  |
|  |  |

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

1. Metaphor / Architecture Specification Template



1. Source Program Listing



*Controller.java*

*package edu.uniandes.ecos.psp21.controller;*

*import edu.uniandes.ecos.psp21.model.Calculate;*

*/\*\**

*\* Clase controladora de la aplicación*

*\**

*\* @version 1*

*\* @author Diego*

*\*/*

*public class Controller {*

*/\*\**

*\* Metodo que hace el llamado a los procesos que hacen los calculos del*

*\* negocio*

*\**

*\* @return*

*\* @throws Exception*

*\*/*

*public String ejecutar() throws Exception {*

*try {*

*StringBuilder st = new StringBuilder();*

*Calculate c = new Calculate();*

*c.calcularX(6D, 0.20, 0.00001, 1D);*

*st.append(calcularRespuesta(c, "Test 1"));*

*Calculate c2 = new Calculate();*

*c2.calcularX(15D, 0.45, 0.00001, 1D);*

*st.append(calcularRespuesta(c2, "Test 2"));*

*Calculate c3 = new Calculate();*

*c3.calcularX(4D, 0.495, 0.00001, 8D);*

*st.append(calcularRespuesta(c3, "Test 3"));*

*return st.toString();*

*} catch (Exception e) {*

*throw e;*

*}*

*}*

*/\*\**

*\* Calcula la respuesta del servlet de acuerdo al resultado de los procesos*

*\**

*\* @param c*

*\* @param titulo*

*\* @return st*

*\*/*

*public String calcularRespuesta(Calculate c, String titulo) {*

*StringBuilder st = new StringBuilder();*

*st.append("<h2>").append(titulo).append("</h1>");*

*st.append("<dl>");*

*st.append("<dt><b>Datos ingresados:</b></dt>");*

*st.append("<dd><b>dof: </b>").append(c.getDof()).append("</dd>");*

*st.append("<dd><b>p esperado:</b>").append(c.getP()).append("</dd>");*

*st.append("<dd><b>e:</b>").append(c.getE()).append("</dd>");*

*st.append("<dt></dt>");*

*st.append("<dt><b>Datos calculados:</b></dt>");*

*st.append("<dd><b>p Calculado: </b>").append(c.getPCalculado()).append("</dd>");*

*st.append("<dd><b>x: </b>").append(c.getX()).append("</dd>");*

*st.append("</dl>");*

*st.append("</br>");*

*return st.toString();*

*}*

*}*

*Calculate.java*

*package edu.uniandes.ecos.psp21.model;*

*/\*\**

*\* Clase que realiza los calculos del negocio*

*\**

*\* @version 1*

*\* @author Diego*

*\*/*

*public class Calculate {*

*private Double dof;*

*private Double p;*

*private Double x;*

*private Double e;*

*private Double pCalculado;*

*/\*\**

*\* Contructor por defecto*

*\*/*

*public Calculate() {*

*}*

*/\*\**

*\* Método que calcula la integral de acuerdo al rango de error*

*\**

*\* @param dof*

*\* @param p*

*\* @param e*

*\* @param xInicial*

*\*/*

*public void calcularX(Double dof, Double p, Double e, Double xInicial) {*

*this.dof = dof;*

*this.p = p;*

*this.e = e;*

*x = xInicial;*

*Double resultado;*

*Double d = x/2;*

*resultado = calcularIntegral(10, dof, x);*

*while (Math.abs(resultado - p) > e) {*

*Double resultadoTmp1 = calcularIntegralAproximada(dof, x + d, e);*

*Double resultadoTmp2 = calcularIntegralAproximada(dof, x - d, e);*

*Double diferenciaTmp1 = resultadoTmp1 - p;*

*Double diferenciaTmp2 = resultadoTmp2 - p;*

*if (Math.abs(diferenciaTmp1) > Math.abs(diferenciaTmp2)) {*

*x = x - d;*

*resultado = resultadoTmp2;*

*} else {*

*x = x + d;*

*resultado = resultadoTmp1;*

*}*

*d = d / 2;*

*}*

*this.pCalculado = aproximar(resultado);*

*}*

*/\*\**

*\* Calcula laintegral usando el metodo de Simpson*

*\**

*\* @param dof*

*\* @param x*

*\* @param e*

*\* @return*

*\*/*

*private Double calcularIntegralAproximada(Double dof, Double x, Double e) {*

*Integer segmentos = 10;*

*Double resultado = 0D;*

*Double resultadoTmp = calcularIntegral(segmentos, dof, x);*

*while (Math.abs(resultado - resultadoTmp) >= e) {*

*resultado = resultadoTmp;*

*segmentos = segmentos + 10;*

*resultadoTmp = calcularIntegral(segmentos, dof, x);*

*}*

*return aproximar(resultado);*

*}*

*/\*\**

*\* Método que calcula la integral de acuerdo a los parámetros ingresados*

*\**

*\* @param segmentos*

*\* @param dof*

*\* @param x*

*\* @return total - Valor de la integral*

*\*/*

*private Double calcularIntegral(Integer segmentos, Double dof, Double x) {*

*Double c1 = (dof + 1) / 2;*

*Double w = x / segmentos;*

*Double xi = 0D;*

*Double total = 0D;*

*Double r1, r2;*

*if (dof % 2 == 0) {*

*r1 = factorialDecimal((dof + 1D), 2D);*

*r2 = factorialEntero(dof / 2);*

*} else {*

*r1 = factorialEntero(((dof + 1D) / 2));*

*r2 = factorialDecimal(dof, 2D);*

*}*

*for (int i = 0; i <= segmentos; i++) {*

*Double v1 = 1 + (Math.pow(xi, 2) / dof);*

*Double v2 = Math.pow(v1, -1 \* c1);*

*Double v3 = r1 / (Math.pow((dof \* Math.PI), 0.5) \* r2);*

*Double v4 = v2 \* v3;*

*Double multiplicador;*

*if (i == 0 || i == segmentos) {*

*multiplicador = 1D;*

*} else if (i % 2 != 0) {*

*multiplicador = 4D;*

*} else {*

*multiplicador = 2D;*

*}*

*Double v5 = (w / 3) \* multiplicador \* v4;*

*xi += w;*

*total += v5;*

*}*

*return total;*

*}*

*/\*\**

*\* Método que aproxima a 5 decimales el valor ingresado*

*\**

*\* @param d*

*\* @return Double valor de la aproximacion*

*\*/*

*private Double aproximar(Double d) {*

*return (Math.round(d \* 100000D) / 100000D);*

*}*

*/\*\**

*\* Calcula el factorial de un número entero*

*\**

*\* @param numero*

*\* @return*

*\*/*

*private Double factorialEntero(Double numero) {*

*Double factorial = 1D;*

*numero--;*

*while (numero != 0) {*

*factorial = factorial \* numero;*

*numero--;*

*}*

*return factorial;*

*}*

*/\*\**

*\* Calcula el factorial de un número fraccionario*

*\**

*\* @param numerador*

*\* @param denominador*

*\* @return*

*\*/*

*private Double factorialDecimal(Double numerador, Double denominador) {*

*Double factorial = 1D;*

*numerador = numerador - 2D;*

*while (numerador >= 1) {*

*factorial = factorial \* (numerador / denominador);*

*numerador = numerador - 2D;*

*}*

*return factorial \* Math.sqrt(Math.PI);*

*}*

*/\*\**

*\* Get del dof del proceso*

*\**

*\* @return dof*

*\*/*

*public Double getDof() {*

*return dof;*

*}*

*/\*\**

*\* Get del x del proceso*

*\**

*\* @return x*

*\*/*

*public Double getX() {*

*return x;*

*}*

*/\*\**

*\* Get del pEsperado*

*\**

*\* @return p*

*\*/*

*public Double getP() {*

*return p;*

*}*

*/\*\**

*\* Get del e*

*\**

*\* @return e*

*\*/*

*public Double getE() {*

*return e;*

*}*

*/\*\**

*\* Get del pCalculado*

*\**

*\* @return pCalculado*

*\*/*

*public Double getPCalculado() {*

*return pCalculado;*

*}*

*}*

*WebView.java*

*package edu.uniandes.ecos.psp21.view;*

*import edu.uniandes.ecos.psp21.controller.Controller;*

*import java.io.IOException;*

*import java.util.logging.Level;*

*import java.util.logging.Logger;*

*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 que maneja las respuestas posibles que genera el servlet*

*\**

*\* @version 1*

*\* @author Diego*

*\*/*

*public class WebView extends HttpServlet {*

*/\*\**

*\* Metodo inicial del servlet*

*\**

*\* @param args*

*\*/*

*public static void main(String[] args) {*

*//Server server = new Server(8380);*

*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 WebView()), "/\*");*

*try {*

*server.start();*

*server.join();*

*} catch (Exception ex) {*

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

*}*

*}*

*/\*\**

*\* Metodo que captura los llamados del doGet*

*\**

*\* @param req*

*\* @param resp*

*\* @throws ServletException*

*\* @throws IOException*

*\*/*

*@Override*

*protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {*

*try {*

*showHome(resp);*

*} catch (ServletException ex) {*

*error(resp, ex);*

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

*} catch (IOException ex) {*

*error(resp, ex);*

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

*}*

*}*

*/\*\**

*\* Metodo que captura los llamados del doPost*

*\**

*\* @param req*

*\* @param resp*

*\* @throws ServletException*

*\* @throws IOException*

*\*/*

*@Override*

*protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException{*

*try {*

*showHome(resp);*

*Controller controller = new Controller();*

*show(resp, controller.ejecutar());*

*} catch (Exception ex) {*

*error(resp, ex);*

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

*}*

*}*

*/\*\**

*\* Inicia el contenido de la pagina del servlet*

*\**

*\* @param req*

*\* @param resp*

*\* @throws ServletException*

*\* @throws IOException*

*\*/*

*private static void showHome(HttpServletResponse resp) throws ServletException, IOException {*

*StringBuilder home = new StringBuilder();*

*home.append("<html>");*

*home.append("<h2>PSP2.1 Program!</h1>");*

*home.append("<h2>Este programa integra num&eacute;ricamente una funci&oacute;n usando la regla de Simpson y calculando din&aacute;micamente el valor de x.</h2>");*

*home.append("<h3>Autor: Diego Lozano</h3>");*

*home.append("<form action=\"calcular\" method=\"post\"> \n<input type=\"submit\" value=\"Calcular\">\n</form> ");*

*home.append("</html>");*

*show(resp, home.toString());*

*}*

*/\*\**

*\* Crea la respuesta de error que se genere en el proceso*

*\**

*\* @param req*

*\* @param resp*

*\* @param error*

*\* @throws ServletException*

*\* @throws IOException*

*\*/*

*private static void error(HttpServletResponse resp, Exception e) throws ServletException, IOException {*

*show(resp, "Error!!!");*

*show(resp, e.getMessage() + ": " + e.getCause());*

*}*

*/\*\**

*\* Crea la respuesta de los resultados de los procesos ejecutados*

*\**

*\* @param req*

*\* @param resp*

*\* @param respuesta*

*\* @throws ServletException*

*\* @throws IOException*

*\*/*

*private static void show(HttpServletResponse resp, String respuesta) throws ServletException, IOException {*

*resp.getWriter().println(respuesta);*

*}*

*}*

*CalculateTest.java*

*package edu.uniandes.ecos.psp21.model;*

*import junit.framework.TestCase;*

*/\*\**

*\**

*\* @author Diego*

*\*/*

*public class CalculateTest extends TestCase {*

*public CalculateTest(String testName) {*

*super(testName);*

*}*

*@Override*

*protected void setUp() throws Exception {*

*super.setUp();*

*}*

*@Override*

*protected void tearDown() throws Exception {*

*super.tearDown();*

*}*

*/\*\**

*\* Test of calcularX1OK method, of class Calculate.*

*\*/*

*public void testCalcularX1OK() {*

*System.out.println("calcularX1OK");*

*Calculate instance = new Calculate();*

*instance.calcularX(6D, 0.20, 0.00001, 1D);*

*assertEquals("Fallo de calculo de variable pCalculada",0.20001, instance.getPCalculado());*

*assertEquals("Fallo de calculo de variable x",0.55340576171875, instance.getX());*

*}*

*/\*\**

*\* Test of calcularX2OK method, of class Calculate.*

*\*/*

*public void testCalcularX2OK() {*

*System.out.println("calcularX2OK");*

*Calculate instance = new Calculate();*

*instance.calcularX(15D, 0.45, 0.00001, 1D);*

*assertEquals("Fallo de calculo de variable pCalculada",0.45, instance.getPCalculado());*

*assertEquals("Fallo de calculo de variable x",1.7530517578125, instance.getX());*

*}*

*/\*\**

*\* Test of calcularX3OK method, of class Calculate.*

*\*/*

*public void testCalcularX3OK() {*

*System.out.println("calcularX3OK");*

*Calculate instance = new Calculate();*

*instance.calcularX(4D, 0.495, 0.00001, 8D);*

*assertEquals("Fallo de calculo de variable pCalculada",0.495, instance.getPCalculado());*

*assertEquals("Fallo de calculo de variable x",4.603515625, instance.getX());*

*}*

*}*

1. Test results

***Test 1***

***Datos ingresados:***

***dof:*** ***6.0***

***p esperado:0.2***

***e:1.0E-5***

***Datos calculados:***

***p Calculado:*** ***0.20001***

***x:*** ***0.55340576171875***

***Test 2***

***Datos ingresados:***

***dof:*** ***15.0***

***p esperado:0.45***

***e:1.0E-5***

***Datos calculados:***

***p Calculado:*** ***0.45***

***x:*** ***1.7530517578125***

***Test 3***

***Datos ingresados:***

***dof:*** ***4.0***

***p esperado:0.495***

***e:1.0E-5***

***Datos calculados:***

***p Calculado:*** ***0.495***

***x:*** ***4.603515625***

