**Operational Specification Template**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student** | Eduardo Sánchez Bautista | **Date** | 03/03/16 |
| **Program** | 6A | **Program #** | 6 |
| **Instructor** | Patricia | **Language** | JAVA |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Scenario Number** | **1** | **User Objective** | Execute the program and get the results | |
| **Scenario Objective** | | Is an overall test of the program, because will be the only interaction with the user, He will push a button to get the expected results from X | | |
| **Source** | **Step** | **Action** | | **Comments** |
| User input | 1 | User push the “*calculate*” button | | The user will use the GUI and the only button present |
|  | 2 | User will compare the expected values against the values retrieved from the program | | The user needs to evaluate the results and get satisfied by them. |

Functional Specification Template

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Student** | | | Eduardo Sánchez Bautista | | **Date** | 03/03/16 |
| **Program** | | | 6A | | **Program #** | 6 |
| **Instructor** | | | Patricia Benavides =) | | **Language** | JAVA |
|  | | | | | | |
| **Class Name** | | CalculationP | | | | |
| **Parent Class** | | void | | | | |
| **Attributes** | | | | | | |
|  | **Declaration** | | | **Description** | | |
|  | S10 : Simpson | | | Simpson object for calculate Simpson with 10 sections. | | |
|  | S20 : Simpson | | | Simpson object for calculate Simpson with 20 sections. | | |
|  | x : double | | | The value x need it for the calculation of P | | |
|  | eVal : double | | | The error value defined for the expected calculations | | |
|  | Dof : double | | | The degree of freedom, need it for the calculation of P | | |
| **Functions** | | | | | | |
|  | **Declaration** | | | **Description** | | |
|  | CalculationP(x, eVal, dof) : void | | | The constructor, will initialize the attributes with the values passed trough the firm. | | |
|  | Calculate() : double | | | The calculation method, will instantiate the Simpson objects, call the P calculation and compare the subtraction form s20 – s10 with the error defined. | | |
|  | getX() : double | | | Get the value X from this object. | | |
|  | setX( double x ): void | | | Set the value of X for this object. | | |
|  | getDof() : double | | | Get the value DOF from this object. | | |
|  | setDof( double dof ): void | | | Set the value of DOF for this object. | | |

|  |
| --- |
|  |
| **Class Name** | | SearchXValue | |
| **Parent Class** | | Null | |
| **Attributes** | | | |
|  | **Declaration** | | **Description** |
|  | P : double | | The P value we are looking to obtain. |
|  | D : double | | The calibrator for the X value |
|  | X : double | | The value x need it for the calculation of P |
|  | Dof : double | | The degree of freedom, need it for the calculation of P |
|  | eCalc : double | | The error calculated from the subtraction of the Pcalculated - Pexpected |
|  | eVal : double | | The error value defined for the expected calculations |
| **Functions** | | | |
|  | **Declaration** | | **Description** |
|  | SearchXValue(double p, dof) : double | | The constructor, will initialize the attributes with the values passed trough the firm. |
|  | Search() : double | | This will perform the search algorithm. Look the flow diagram in the conceptual design. |
|  | getX() : double | | Get the value X from this object. |
|  | setX( double x ): void | | Set the value of X for this object. |
|  | getDof() : double | | Get the value DOF from this object. |
|  | setDof( double dof ): void | | Set the value of DOF for this object. |
|  | getP (): double | | Get the value P from this object. |
|  | setP (double dof): void | | Set the value of P for this object. |

|  |
| --- |
|  |
|  |
| **Class Name** | | ListenerForButton | |
| **Parent Class** | | Null but will implement the ActionListener Interface | |
| **Attributes** | | | |
|  | **Declaration** | | **Description** |
|  | Searcher : SearchXValue | | Simpson object for calculate Simpson with 10 sections. |
| **Functions** | | | |
|  | **Declaration** | | **Description** |
|  | actionPerformed(e : ActionEvent): void | | This will listen for when the calculate button will be touched, then will instantiate the searcher object and will modify the values of the JTable for the X values calculated |

|  |
| --- |
|  |
| **Class Name** | | GUI | |
| **Parent Class** | | JFrame | |
| **Attributes** | | | |
|  | **Declaration** | | **Description** |
|  | buttonLIstener : LIstenerForButton | | The lisener for the calculate button that will eventually perform the whole calculatioins |
|  | Columns : String[] | | The columns names for the JTable |
|  | Data : Object[] | | The body of the JTable |
|  | Table : JTable | | The table |
|  | Calculate : JButton | | The button that will initiate the calculations |
|  | Test : Jlabel | | A label for the word test |
|  | expectedLabel : JLabel | | A label for the word Expected |
|  | actualLabel : JLabel | | A label for the word Actual |
|  | scrollPane : JScrollPane | | The scroll panel will contain the table |
|  | thePanel : JPanel | | The panel will contain the whole app |
| **Functions** | | | |
|  | **Declaration** | | **Description** |
|  | GUI() : void | | The constructor of the GUI |
|  | Public static main(String[] args) | | The main function will perform as sandbox |
|  |
|  |
|  |
|  |
|  |

State Specification Template

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Student | | Eduardo Sánchez Bautista | | | Date | | 02/03/2016 |
| Program | | 6 A | | | Program # | | 6 |
| Instructor | | Patricia Benavides | | | Language | | JAVA |
|  | | |  | | | | |
| **State Name** | | | | **Description** | | | |
| Start | | | | The program will be started and waiting for the calculation to begins | | | |
| Calculating | | | | The button *calculate* will be pressed and then the program will execute the search algorithm | | | |
| Calculated | | | | The Values were calculated and now will be showed in the GUI | | | |
| Error | | | | The values are incorrect | | | |
| Success | | | | The values correlate with the expected values | | | |
| **Function/Parameter** | | | | **Description** | | | |
| Button listening | | | | The program will be waiting for the click on he only button. | | | |
| Close program | | | | The program will be listening if the program is shut down. | | | |
| **States/Next States** | | | | **Transition Condition** | | **Action** | |
| Start | | | |  | |  | |
| Calculated | | | The button will be pressed | | Button actioned | |
| Calculated | | | |  | |  | |
| Error | | | The values were calculated | | Update of the GUI with the values | |
| Success | | | The values were calculated | | Update of the GUI with the values | |

**Logic Specification Template**

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Eduardo Sánchez Bautista | Date | 02/03/2016 |
| Program | 6 A | Program # | 6 |
| Instructor | Patricia Benavides | Language | JAVA |

|  |  |
| --- | --- |
| **Design** | [Conceptual design](https://docs.google.com/a/upaep.edu.mx/document/d/1w8OxHkgRijiDPHG0MhAqFEJjl4882BG2s7XWLATxZf8/edit?usp=sharing) |
| **References** | [Requirement Statements](https://docs.google.com/a/upaep.edu.mx/document/d/1ufkxTLYjWWgfxpwyUVknH4ewZO3oEH_alQMeH9YOXkM/edit?usp=sharing) |
|  | Operational specification |
|  | Functional specification |
|  | States specifications |

|  |  |
| --- | --- |
| **Parameters** | p : The value of what we want to know the x value |
|  | x : the value we define initial like 1 |
|  | d : the value for calibrate x we initial set like 0.5 |
|  | eVal : The error of acceptance for the test case search x value |
|  |  |

|  |
| --- |
| Search the X value for the P value defined by the test case |
| Start by calculating the Simpson p for x = 1 => pCalculated == original ? **print x && break**: **continue**; |
| This calculated P value is greater or smaller than the test case P value? |
| If **yes** x = x + d |
| If **no** x = x – d |
| We recalculate the Simpson p value again with the new x |
| Now we calculate the error from subtracting pCalculated – testCaseP |
| This error calculated is the same as the eVal? |
| If **yes** **we print the X** we used |
| If **no** we have to calibrate *D* |
| When calibrating *D* first we have to check if the errorCalculated have the same sign as the eVal |
| If **yes** we left *D* the same |
| If **no**  we transform *D = D / 2* |
| We have to ensure that *D* will never be 0, if this happen we set it as 2 |
| We repeat the cycle form the line **3** until we get X |
|  |