*Logic Specification Template*

|  |  |  |  |
| --- | --- | --- | --- |
| **Student** | Iker Arbulu Lozano | **Program #** | 4 |

|  |  |
| --- | --- |
| **Class Name** | Programa1 |

|  |  |
| --- | --- |
| **Design** | Todos los OST |
| **References** |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Method Name** | Main |

|  |  |
| --- | --- |
| **Parameters** | Args[]: String |
|  |  |
|  |  |
|  |  |
|  |  |

|  |
| --- |
| Abrir el buffer de lectura para el usuario |
| Crear la variable de tipo CalculadorE |
| Crear la variable de tipo flotante para la X |
| Crear la variable de tipo entero para los DOF |
| Desplegar (“Dame la X”) |
| If(Pattern.matches(“\\d+(\\.\\d+)?”,currentLine) |
| If(Float.parseFloat(currentLine)>=0) |
| Asignar la entrada del usuario a la X |
| Else |
| Desplegar(“X incorrecta”) |
| System.exit(0) |
| Else |
| Desplegar(“X incorrecta”) |
| System.exit(0) |
| Desplegar (“Dame los DOF”) |
| If(Pattern.matches(“[\\d+](file:///\\d+)”,currentLine) |
| If(Int.parseInt(currentLine)>0) |
| Asignar la entrada del usuario a los DOF |
| Else |
| Desplegar(“DOF incorrectos”) |
| System.exit(0) |
| Else |
| Desplegar(“DOF incorrectos”) |
| System.exit(0) |
| Crea calculadorE(X,DOF) |
| calculadorE.print() |
| Cerrar el buffer de lectura para el usuario |

|  |  |
| --- | --- |
| **Class Name** | CalculadorE |

|  |  |
| --- | --- |
| **Design** | OST 1 |
| **References** |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Method Name** | CalculadorE |

|  |  |
| --- | --- |
| **Parameters** | fX: float |
|  | iDof: int |
|  |  |
|  |  |
|  |  |

|  |
| --- |
| This.fX = fX |
| This.iDof = iDof |
| CalculaP() |

|  |  |
| --- | --- |
| **Class Name** | CalculadorE |

|  |  |
| --- | --- |
| **Design** | OST 1 |
| **References** |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Method Name** | CalculaP |

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

|  |
| --- |
| iNum\_seg = 10 |
| cpCalcP = new CalculadorP(fX,iDof,iNum\_seg) |
| fP1 = cpCalcP.CalculaP() |
| iNum\_seg \*=2 |
| cpCalcP.setNum\_seg(iNum\_seg) |
| fP2 = cpCalcP.calculaP() |
| While(abs(fP1 – fP2) >= fE) |
| fP1 = fP2 |
| iNum\_seg\*=2 |
| cpCalcP.setNum\_seg(iNum\_seg) |
| fP2 = cpCalcP2.calculaP() |
| End while |
| fP = fP2 |

|  |  |
| --- | --- |
| **Class Name** | CalculadorE |

|  |  |
| --- | --- |
| **Design** | OST 1 |
| **References** |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Method Name** | Print |

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

|  |
| --- |
| System.out.println(“x = ”+ String.*format*("%.5f", fX)) |
| System.out.println(“dof = ”+ iDof) |
| Sytem.out.println(“p = ”+ String.format(“%.5f”, fP)) |

|  |  |
| --- | --- |
| **Class Name** | CalculadorP |

|  |  |
| --- | --- |
| **Design** | OST 1 |
| **References** |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Method Name** | CalculaP |

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

|  |
| --- |
| Float fW |
| fW = (float)(fX/iNum\_seg) |
| fSumP = |
| For(float fXi=fW; fXi<fX; fXi+=fW) |
| If(fXi%2 ==0) |
| fSumP += w/3\*4\*calculaTDist(fXi) |
| Else |
| fSumP += w/3\*2\*calculaTDist(fXi) |
| End For |
| fSumP += w/3\*calculaTDist(fX) |
| Return fSumP |

|  |  |
| --- | --- |
| **Class Name** | CalculadorP |

|  |  |
| --- | --- |
| **Design** | OST 1 |
| **References** |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Method Name** | CalculaTDist |

|  |  |
| --- | --- |
| **Parameters** | fXi: float |
|  |  |
|  |  |
|  |  |
|  |  |

|  |
| --- |
| Float fT |
| fT = pow((1 + pow(fXi,2)/iDof),-((iDof+1)/2)) |
| fT \***= (calculaGamma((iDof+1)/2)/((pow(iDof**\*pi()),0.5)\*calculaGamma(iDof/2) |
| Return fT |

|  |  |
| --- | --- |
| **Class Name** | CalculadorP |

|  |  |
| --- | --- |
| **Design** | OST 1 |
| **References** |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Method Name** | CalculaGamma |

|  |  |
| --- | --- |
| **Parameters** | fNum: float |
|  |  |
|  |  |
|  |  |
|  |  |

|  |
| --- |
| fGamma = 1 |
| fNum-- |
| If(fNum%1 == 0) |
| While(fNum>1) |
| fGamma \*=fNum |
| fNum-- |
| EndWhile |
| Else |
| While(fNum>=0.5) |
| fGamma\*=fNum |
| fNum-- |
| End While |
| fGamma\*=sqrt(pi) |
| End If |
| Return fGamma |