Logic Specification Template

Student José Alberto Esquivel Patiño **Program #** 5

Class Name IOHandler

Method Name readValue

Parameters sPrompt : String

sErrorMessage : String

patValidStructure : Pattern

declare sValue
declare matValidator
assign patValidStructure.matcher() to matValidator
print sPrompt
read sValue
trim sValue
while sValue does NOT match with matValidator
print sErrorMessage
print sPrompt
read sValue
return sValue

Class Name	AreaUnderTDistribution
Method Name	main
Parameters	

declare and initialise variable areXCalculator of type AreaUnderTDistribution
declare and initialise variable ioHandler of type IOHandler
call readValue on ioHandler with parameters: “Introduce el valor del área para la cuál se calculará x: (debe de ser numérico real, entre 0 y 0.5)”, sINVALID_PVALUE, Pattern.compile("((0+)?(\\.(([0-4]\\d*)|50*)))|((0+)(\\.(([0-4]\\d*)|50*))?)")
parse the returned value to double
assign the returned value to class variable dP on areXCalculator
call readValue on ioHandler with parameters: “Introduce el valor de los grados de libertada dof: (debe de ser numérico entero y mayor a 0)”, sINVALID_INTEGER, Pattern.compile(“\\d*[1-9]\\d*")
parse the returned value to Integer
assign the returned value to class variable iDof on areXCalculator
call the class function calculateX on areXCalculator
print areXCalculator

Class Name	AreaUnderTDistribution
Method Name	gamma
Parameters	dX : double

if dX is 1
return 1
else if dX is 1/2
return square root of PI
else
return (dX - 1) * gamma(dX - 1)

Class Name	AreaUnderTDistribution
Method Name	tStudent
Parameters	dX : double

return (gamma((iDof + 1) / 2) / ((dof*PI)^(1/2) * gamma(dof / 2))) * (1 + (dX^2/iDof)) ^ ((iDof + 1)/(-2))

Class Name	AreaUnderTDistribution
Method Name	calculate
Parameters	

declare iNumSeg and initialise with 8
declare dW and initialise with dX / iNumSeg
declare dE and initialise with 0.0000001
declare dPreviousP
call simpson with parameters : dX, dW, iNumSeg
assign returned value to dP
do the following
assign dP to dPreviousP
double the value of iNumSeg
assign dX / iNumSeg to dW
call simpson with parameters : dX, dW, iNumSeg
assign returned value to dP
while the absolute value of (dP - dPreviousP) > dE

Class Name	AreaUnderTDistribution
Method Name	simpson
Parameters	dX : double dW : double iNumSeg - 1 : Integer

declare dSum4W and initialise with 0
declare dSum2W and initialise with 0
declare iCont and initialise with 1
while iCont is less than or equal iNumSeg - 1
dSum4W += 4 * tStudent(iCont * dW)
assign iCont + 2 to iCont
assign 2 to iCont
while iCont is less than or equal iNumSeg - 2
dSum2W += 2 * tStudent(iCont * dW)
assign iCont + 2 to iCont
return (dW / 3) * (tStudent(0) + dSum4W + dSum2W + tStudent(dX))

Class Name	AreaUnderTDistribution
Method Name	toString

declare variable sFormat
assign “p = %.5f\ndof = %d\nx = %.5f” to sFormat
return sFormat.format(dP, iDof, dX);

Class Name	AreaUnderTDistribution
Method Name	calculateX
Parameters	

declare dTargetP and initialise with dP
declare dError and initialise with 0.00000001
declare dD and initialise with 0.5
assign 1.0 to dX
call calculate()

declare dPreviousError and initialise with dTargetP - dP
while the absolute value of dTargetP - dP is > than dError
dX = (dP > dTargetP) ? dX - dD : dX + dD
call calculate()
dD = (dPreviousError * (dTargetP - dP) < 0) ? dD / 2 : dD;
dPreviousError = dTargetP - dP