Tp algo 3 pseudo code

STRUCT Equation

PARAM entier a, b, c

STRUCT Solution solution

STRUCT Solution

SolutionType type

PARAM réel x0, x1, x2

ENUM SolutionType (NONE,ONE,TWO)

FONCTION showSolution(PARAM Solution solution)

SWITCH solution.type

CASE NONE

PRINT "Il n'y a aucune solution"

CASE ONE

PRINT "Il y a une solution : ", solution.x0

CASE TWO

PRINT "Il y a deux solutions : ", solution.x1, " et solution.x2

DEFAULT

PRINT "Type de solution non reconnu"

FIN SWITCH

FIN FONCTION

FONCTION resolveOne(PARAM entier b,c)

RETOURNER -c / b

FIN FONCTION

FONCTION resolveTwo(PARAM entier a, b, c, VAR Solution \*solution)

VAR delta <- (b \* b) - 4 \* a \* c

SI delta > 0

solution->type <- TWO

solution->x1 <- (-b - sqrt(delta)) / (2.0 \* a)

solution->x2 <- (-b + sqrt(delta)) / (2.0 \* a)

SINON SI delta == 0

solution->type <- ONE

solution->x0 <- (-b / (2.0 \* a))

SINON

solution->type <- NONE

FIN SI

FIN FONCTION

FONCTION resolve(PARAM Equation \*equation)

SI equation->a != 0 ALORS

resolveTwo(equation->a, equation->b, equation->c, &equation-> solution)

SINON

equation->solution.type <- ONE

equation->solution.x0 <- resolveOne(equation->b, equation->c)

FIN SI

FIN FONCTION

FONCTION decode(PARAM chaîne equa, VAR Equation \*equation)

SI sscanf(equa, "%dx` +%dx +%d", &equation->a, &equation->b, &equation->c) == 3

resolve(equation)

RETOURNER equation->solution

SINON SI sscanf(equa, "%dx +%d", &equation->b, &equation->c) == 2

equation->a = 0

resolve(equation)

RETOURNER equation->solution

SINON SI sscanf(equa, "%dx` +%d", &equation->a, &equation->b) == 2

equation->c = 0

resolve(equation)

RETOURNER equation->solution

SINON SI sscanf(equa, "%dx` +%", &equation->a, &equation->c) == 2

equation->b = 0

resolve(equation)

RETOURNER equation->solution

SINON

PRINT "Format invalide: ", equa

equation->solution.type = NONE

RETOURNER equation->solution

FIN SI

FIN FONCTION

FONCTION test()

// Test resolve 1 : 2x-4=0

PARAM entier b1 = 2, c1=4

VAR solution1 = resolveOne(b1, c1)

// Test resolve 2 delta+: -3x + 6 + 2 = 0

PARAM entier a2 = -3,b2 = 6,c2 = 2

STRUCT Solution solution2

resolveTwo(a2, b2, c2, &solution2)

showSolution(solution2)

// Test resolve2 delta=0: -3x + 6 + 2 = 0

PARAM entier a3 = 2,b3 = 4,c3 = 2

STRUCT Solution solution3

resolveTwo(a3, b3, c3, &solution3)

showSolution(solution3)

// Test resolve2 delta-: 3x + 2 + 2 = 0

PARAM entier a4 = 3,b4 = 2, c4 = 2

STRUCT Solution solution4

resolveTwo(a4, b4, c4, &solution4)

showSolution(solution4)

// Test resolve

PARAM entier a5 = 3,b5 = 5, = 1

PRINT "resolveone : "

STRUCT Equation equation5

equation5.a = 0

equation5.b = b5

equation5.c = c5

resolve(equation5)

showSolution(equation5.solution)

PRINT "resolvetwo : "

STRUCT Equation equation6

equation6.a = a5

equation6.b = b5

equation6.c = c5

resolve(equation6)

showSolution(equation6.solution)

STRUCT Equation equation7

STRUCT Equation equation8

STRUCT Equation equation9

STRUCT Equation equation10

STRUCT Equation equation11

VAR chaîne equa1[150]

decode("5x` +4x +1", equation7) // cas pour ax^2 +bx + c

decode("3x +-9", equation8) // cas pour bx + c

decode("2x` +1", equation10) // cas pour ax^2 + c

decode("2x` +1x", equation11) // cas pour ax^2 + bx

PRINT "cas pour ax^2 +bx + c : "

showSolution(equation7.solution)

PRINT "cas pour bx + c : "

showSolution(equation8.solution)

PRINT "cas pour ax^2 + c : "

showSolution(equation10.solution)

PRINT "cas pour ax^2 + bx : "

showSolution(equation11.solution)

fgets(equa1, 150, stdin)

decode(equa1, equation9)

showSolution(equation9.solution)

FIN FONCTION

FONCTION main()

test()

FIN FONCTION