UNIVERSIDAD DE ANTIOQUIA

FACULTAD DE INGENIERIA

DEPARTAMENTO DE INGENIERÍA DE SISTEMAS

CUARTO EXAMEN LÓGICA II (2020/1)

Nombre \_\_\_\_Joan Manuel Muñoz Monroy\_\_\_\_\_\_\_\_\_\_\_ Cédula \_\_\_\_\_\_\_\_\_\_\_\_\_

Directrices: Retornar la solución del examen en un archivo tipo Word, tipo texto, tipo excell o tipo PowerPoint. Junto con la solución retornar también el archivo del enunciado. Si resolvió el examen en el archivo del enunciado, mejor.

El correo que envía debe tener el **asunto**: cuartoExamenLogicaIIxxxx siendo xxxx el nombre del estudiante que lo presenta. Adicionalmente, el nombre del archivo en el cual envía la solución debe ser lo mismo que colocó en el **asunto.**

1. Elabore algoritmo para evaluar una expresión en infijo.

2. Haga seguimiento detallado al siguiente algoritmo especificando claramente lo que escribe y el orden en que lo escribe.

void pk09(int m, int o, int d, int a)

if (m > 0)

pk09(m - 1, d, o, a)

escriba(a, d, o)

pk09(m - 1, a, o, d)

fin

fin

// al ser llamado pk09(3, 2, 1, 3)

Solución

double evaInfi(hilera S)

stack pila1, pila2

hilera x, opdor

int op1, op2

boolean estado = true

x = S.siguienteToken()

while(x != null)

if(x in operadores)then

pila2.apilar(x)

else

pila1.apilar(x)

x = S.siguienteToken()

if(!pila2.esVacia()) then

while(estado && x != null)do

if(x in operadores)then

if(pfp(x) >= pfp(pila2.cima()))

pila2.apilar(x)

else

estado = false

end(if)

else

pila1.apilar(x)

end(if)

x = S.siguienteToken()

end(while)

while(!pila2.esVacia())do

op1 = pila1.desapilar()

op2 = pila1.desapilar()

opdor = pila2.desapilar()

casos de opdor:

"^": res = op2 ^ op1

"\*": res = op2 \* op1

"/": res = op2 / op1

"%": res = op2 % op1

"-": res = op2 - op1

"+": res = op2 + op1

fin(casos)

pila1.apilar(res)

end(while)

pila2.apilar(x)

end(if)

end(if)

x = S.siguienteToken()

end(while)

return pila1.cima()

fin(evaInfi)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| L2 | 0 | 2 | 1 | 3 |
| L1 | 0 | 3 | 1 | 2 |
| L2 | 1 | 1 | 3 | 2 |
| L2 | 0 | 1 | 2 | 3 |
| L1 | 0 | 3 | 2 | 1 |
| L1 | 1 | 2 | 3 | 1 |
| L2 | 2 | 3 | 2 | 1 |
| L2 | 0 | 2 | 3 | 1 |
| L1 | 0 | 1 | 3 | 2 |
| L2 | 1 | 3 | 1 | 2 |
| L2 | 0 | 3 | 2 | 1 |
| L1 | 0 | 1 | 2 | 3 |
| L1 | 1 | 2 | 1 | 3 |
| L1 | 2 | 1 | 2 | 3 |
| L0 | 3 | 2 | 1 | 3 |
| DR | m | o | d | a |

al ser llamado pk09(3, 2, 1, 3)

L0

void pk09(int m, int o, int d, int a)

if (m > 0)

pk09(m - 1, d, o, a)

L1 escriba(a, d, o)

pk09(m - 1, a, o, d)

L2 fin

fin

Lo que el algoritmo imprime:

3,1,2

3,2,1

2,1,3

3,1,2

1,3,2

1,2,3

2,3,1