

Introducción a la programación

Actividad en Cose

Taller 13

```

1. Inicio
2. función principal()
3. mensaje , x(50){[a-zA-Z]{0,9},{BS}}]
4. mensaje --> descomponer()
5. Escribir mensaje
6. Fin función principal

7. función descomponer(); x
8. K, d [0,n]
9. K, i, d [0,n]
10. j, i [0,n] <= 0
11. cadena , x(50){[a-zA-Z]{0,9},{BS}}]
12. residuo , i [0, n]

13. << "Ingrese el numero a descomponer: "
14. >> Leer K
15. K1 <- K

16. Si (K1 >= 0) entonces
17. Mientras (K1 > 0) entonces
18.     residuo <- obtenerResiduo(K1)
19.     cadena <- cadena + residuo + "* 10 elevado
           a lo potencia" + j
20.     K1 <- K1/10
21.     j = j + 1
22. De lo contrario
23.     >> "Error, cantidad no valida"
24. Fin Si
25. Fin Mientras
26. retornar cadena
27. Fin función descomponer()

28 función obtenerResiduo(o,i): i
29. z, i [0-n]
30. z <- * MOD 10
31. retornar z
32 Fin función obtenerResiduo()
33. Fin

```

Introducción a la programación

Actividad en Cose

Taller 13

1. Inicio
2. función principal ()
3. mensaje , x(50)[{a-z},{A-Z},{0,9},{BS}]
4. mensaje L--descomponer ()
5. Escribir mensaje
6. Fin función principal
7. función descomponer () ; x
8. K,d [0,n]
9. k,i,d [0,n]
10. j,i [0,n] <= 0
11. cadena , x(50)[{a-z},{A-Z},{0,9},{BS}]
12. residuo , i [0,n]
13. << "Ingrese el numero a descomponer:"
14. >> Leer R
15. K, <= K
16. Si (K>=0) entonces
17. Mientras (K, >0) entonces
18. residuo <- obtenerResiduo (K)
19. cadena <- cadena + residuo + "* 10 elevado
a la potencia" + j
20. K, <- K/10
21. j = j + 1
22. De lo contrario
23. >> "Error, cantidad no valida"
24. Fin Si
25. Fin Mientras
26. retornar cadena
27. fin función descomponer ()
28. función obtenerResiduo (o,i) : i
29. z, i [0-n]
30. z <- * MOD 10
31. retornar z
32. Fin función obtenerResiduo ()
33. Fin