FUNDAMENTOS DE PROGRAMACIÓN

Inicio Clase 10

Profesor: Carlos Díaz

Clase 10: Recursividad

- Definición de recursividad
- Ejercicios

Definición de recursividad

- La recursividad es la propiedad que posee una función de permitir que dicha función pueda llamarse así misma, bien directamente, o bien a través de otra función.
- Toda función recursiva tiene un punto de retorno, sin el cual se llamaría eternamente a si misma.

EJEMPLO *Función recursiva que calcula el factorial de un número n > 0.*

Es conocido que `5! = 5.4.3.2.1 = 5.4! = 120`. Es decir `n! = n.(n - 1)!`
Así pues, la definición recursiva de la función factorial es la siguiente:

```
Factorial(n) = n*Factorial(n-1) si n > 0
Factorial(n) = 1   si n = 0
```

PUNTO DE RETORNO

Ejemplo: Factorial de un número

```
#include <iostream>
using namespace std;
long factorial (int n){
        if (n>0)
                return n*factorial(n-1);

        if (n==0)
                return 1;
}

int main(){
        cout<<factorial(12);
        return 0;
}
```

Contador recursivo

- Escriba un programa que cuente del 1 al 5 recursivamente.

```
#include <iostream>
using namespace std;
void contar (int n){
        if (n==1){
                cout<<n<<endl;
                return;
        }
        contar(n-1);
        cout<<n<<endl;
        return;
}
```

```
int main(){
        contar(5);
        return 0;
}
```

La función de Fibonacci recursiva

- Los números de Fibonacci quedan definidos por las ecuaciones:

```
#include <iostream>
using namespace std;
long Fibo (int n){
        if (n==0 || n==1)
                return n;
        else
                return Fibo(n-1)+Fibo(n-2);
}

int main(){
        for (int n=0;n<=8;n++)
        cout<<Fibo(n)<<endl;
        return 0;
}
```

$$f_0 = 0$$

$$f_1 = 1$$

$$f_n = f_{n-1} + f_{n-2}$$

Esto produce:

$$f_2 = 1$$

$$f_3 = 2$$

$$f_4 = 3$$

$$f_5 = 5$$

$$f_6 = 8$$

$$f_7 = 13$$

$$f_8 = 21$$

Ejercicio 1

- Función recursiva que calcula la suma de los cuadrados de los N primeros números positivos.

$$suma(N) = \begin{cases} 1 & si \quad N = 1 \\ N^2 + suma(N-1) & en\ caso\ contrario \end{cases}$$

Ejercicio 2

- Escribir una función recursiva que calcule la función de Ackermann definida de la siguiente forma:

$$A(m, n) = n + 1 \qquad si \quad m = 0$$
$$A(m, n) = A(m - 1, 1) \qquad si \quad n = 0$$
$$A(m, n) = A(m - 1, A(m, n - 1)) \qquad si \quad m > 0, \ y \ n > 0$$

Ejercicio 3

- Ingrese por teclado el número n de términos, calcule y muestre el valor de la siguiente fracción continua.

$$2+\cfrac{2}{2+\cfrac{3}{3+\cfrac{4}{4+\cfrac{5}{5+\cfrac{6}{6+\cfrac{7}{7+\cdots}}}}}}$$

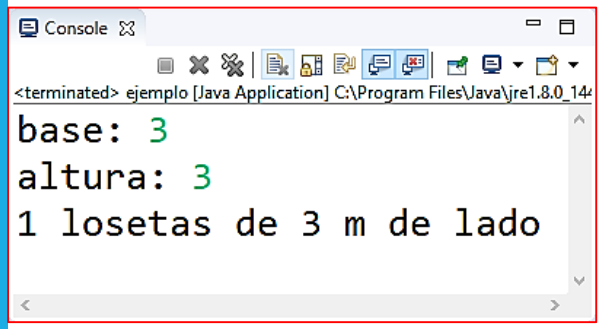
- Ejemplo: Para **n = 4** términos.

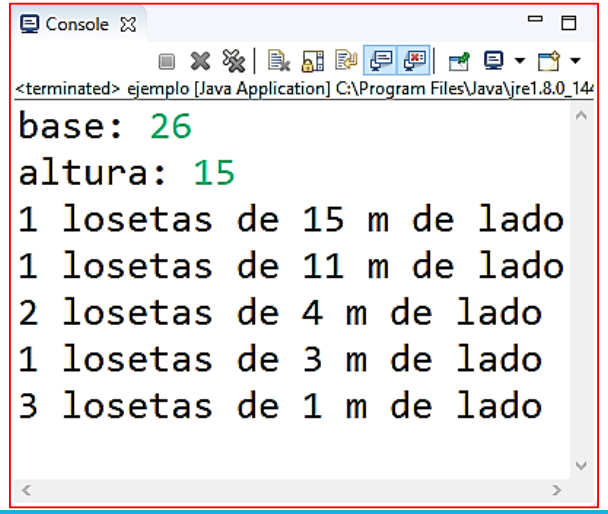
$$a_1+\cfrac{2}{a_2+\cfrac{3}{a_3+\cfrac{4}{a_4}}}=2+\cfrac{2}{2+\cfrac{3}{3+\cfrac{4}{4}}}=\cdots$$

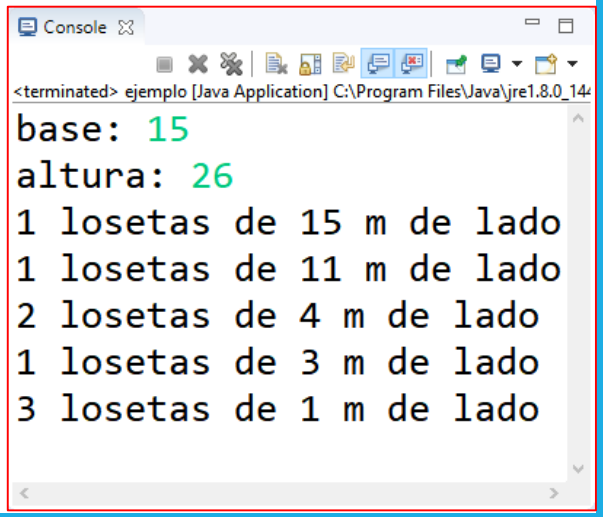
Ejercicio 4

- Se desea colocar losetas en una habitación rectangular utilizando exclusivamente losetas cuadradas de cualquier medida, no necesariamente iguales. Calcule la mínima cantidad de losetas necesarias, mostrando su medida.
- La menor medida posible es 1 m.
- Para hacerlo más interesante, no puede usar ninguna sentencia repetitiva, es decir: While, Do-While ni For.

Ejemplos:







FUNDAMENTOS DE PROGRAMACIÓN

Fin Clase 10

Profesor: Carlos Díaz