

Universidad Autónoma de Zacatecas

Unidad Académica de Ingeniería Eléctrica

Programa Académico de Ingeniería de Software.

Trabajo parcial 2

Nombre de la Práctica Recursión.

Nombre de la carrera Ingeniería de Software

Nombre de la materia Estructuras de Datos

Nombre del alumno Jesús Manuel Juárez Pasillas

Nombre del docente Aldonso Becerra Sánchez

Fecha: 19/10/2021

Trabajo parcial 2: Recursión.

Introducción:

La recursión es de gran utilidad en la programación, con esta podemos llamar a la misma función en la que se esta y con esto obtener diferentes resultados dependiendo el valor de cada uno de sus parámetros. Con esto podemos hacer un bucle en el que se tenga un caso base con el cual parar el bucle y al menos una condición con la cual se pueda dar la recursión.

Desarrollo:

En este trabajo parcial se pide realizar un desglose, el método y un corrimiento con pilas de un problema planteado, todo esto hecho a mano.

Para esto se analizo el problema el cual es una función:

$$A(m,n) = \left\{ egin{array}{ll} n+1, & ext{si } m=0; \ A(m-1,1), & ext{si } m>0 ext{ y } n=0; \ A(m-1,A(m,n-1)), & ext{si } m>0 ext{ y } n>0 \end{array}
ight.$$

De esto se concluye que son tres condiciones diferentes, siendo la primera el caso base (cuando m=0). Teniendo esto en cuenta se pide realizar el desglose de esta función teniendo los valores de m y n, los cuales son libres. El desglose con un número alto es demasiado extenso por lo que se eligieron los valores de m=2 y n=2 y con esto sacar el siguiente desglose:

		A(2,2) €7	Resultado final = 7			
		A(2-1,A(2,2-1)=7)				
	A(1-1,A(1,5-1))=7					
~			A(2-1,A(2,1-1)) = 5			
6+1=7						
	A(1-1,A(1,4-1))∈6	A(1-1,A(1,3-1))=5)		A(2-1,1)≢3	
					· · · · · · · · · · · · · · · · · · ·	
5+1=6		4+1=5	A(1-1,A(1,2-1))€4		A(1-1,A(1,1-1))+3)
	A(1-1,A(1,3-1))€5					
	7		3+1€4	A(1-1,A(1,1-1))=3	2+1 = 3	A(1-1,1)€2
	4+1=5	A(1-1,A(1,2-1)) = 4)	~~~~		6
				2+1=3	A(1-1,1)=2	1+1 =(2)
		3+1=4	A(1-1,A(1,1-1))∈3			
					1+1=2	
			2+1 =3	A(1-1,1)€2		
				1+1(=2)		

El resultado final de este desglose es 7 teniendo como parámetros m=2 y n=2.

Después se pidió realizar el método con el que se resolvería esta función:

```
public static int A(int m, int n){
   if(m > 0 && n == 0){
      return A(m-1,1);
   }else if(m > 0 && n > 0){
      return A(m-1,A(m,n-1));
   }else{
      return n+1;
   }
}
```

Este método contiene las 3 condiciones con las cuales 2 de ellas sigue siendo recursivo y la tercera es el caso base con el cual se rompe la recursión.

Por último, se pidió realizar el corrimiento con pilas del método realizado, este corrimiento quedaría de la siguiente manera:

```
public class Funcion{
   public static void main(String[] args) {
        A(m,n);
    }
}

public static int A(int m, int n){
   if(m > 0 && n == 0){
        return A(m-1,1);
   }else if(m > 0 && n > 0){
        return A(m-1,A(m,n-1));
   }else{
        return n+1;
   }
}
```

Pila	Pila	Pila	Pila	Pila
			1(0,0),0	A(1,1):5
		A(2,1):5	A(2,0):3 A(2,1):5	A(2,0):3 A(2,1):5
	A(2,2):5	A(2,2):5	A(2,2):5	A(2,2):5
main():3	main():3	main():3	main():3	main():3

Pila		Pila	Pila	Pila
A(1,0):3 A(1,1):5 A(2,0):3 A(2,1):5 A(2,2):5 main():3	A(1,1):5 A(2,0):3 A(2,1):5 A(2,2):5 main():3	A(1,1):5 A(2,0):3 A(2,1):5 A(2,2):5 main():3	A(2,0):3 A(2,1):5 A(2,2):5 main():3	A(2,1):5 A(2,2):5 main():3
Pila	Pila	Pila	Pila	Pila
A(2,1):5 A(2,2):5 main():3	A(1,3):5 A(2,1):5 A(2,2):5 main():3	A(1,2):5 A(1,3):5 A(2,1):5 A(2,2):5 main():3	A(1,1):5 A(1,2):5 A(1,3):5 A(2,1):5 A(2,2):5 main():3	A(1,0):3 A(1,1):5 A(1,2):5 A(1,3):5 A(2,1):5 A(2,2):5 main():3
Pila	Pila	Pila	Pila	Pila
A(1,1):5 A(1,2):5 A(1,3):5 A(2,1):5 A(2,2):5 main():3	A(1,1):5 A(1,2):5 A(1,3):5 A(2,1):5 A(2,2):5 main():3	A(1,2):5 A(1,3):5 A(2,1):5 A(2,2):5 main():3	A(1,2):5 A(1,3):5 A(2,1):5 A(2,2):5 main():3	A(1,3):5 A(2,1):5 A(2,2):5 main():3
Pila	Pila	Pila	Pila	Pila
A(1,3):5 A(2,1):5 A(2,2):5 main():3	A(2,1):5 A(2,2):5 main():3	A(2,2):5 main():3	A(1,5):5 A(2,2):5 main():3	A(1,4):5 A(1,5):5 A(2,2):5 main():3

Pila	Pila	Pila	Pila	Pila
Fila	Fila	Fila		Fila
A(1,3):5 A(1,4):5 A(1,5):5 A(2,2):5 main():3	A(1,2):5 A(1,3):5 A(1,4):5 A(1,5):5 A(2,2):5 main():3	A(1,1):5 A(1,2):5 A(1,3):5 A(1,4):5 A(1,5):5 A(2,2):5 main():3	A(1,0):3 A(1,1):5 A(1,2):5 A(1,3):5 A(1,4):5 A(1,5):5 A(2,2):5 main():3	A(1,1):5 A(1,2):5 A(1,3):5 A(1,4):5 A(1,5):5 A(2,2):5 main():3
Pila	Pila	Pila	Pila	Pila
A(1,1):5 A(1,2):5 A(1,3):5 A(1,4):5 A(1,5):5 A(2,2):5 main():3	A(1,2):5 A(1,3):5 A(1,4):5 A(1,5):5 A(2,2):5 main():3	A(1,2):5 A(1,3):5 A(1,4):5 A(1,5):5 A(2,2):5 main():3	A(1,3):5 A(1,4):5 A(1,5):5 A(2,2):5 main():3	A(1,3):5 A(1,4):5 A(1,5):5 A(2,2):5 main():3
Pila	Pila	Pila	Pila	Pila
A(1,4):5 A(1,5):5 A(2,2):5 main():3	A(1,4):5 A(1,5):5 A(2,2):5 main():3	A(1,5):5 A(2,2):5 main():3	A(1,5):5 A(2,2):5 main():3	A(2,2):5 main():3
Pila	Pila			
main():3		Fin del programa	a, retornando 7 co	omo resultado final.

Conclusión:

La recursión es una técnica de programación bastante útil a la hora de tener valores no muy grandes ya que la recursividad requiere de mucha memoria y tiempo de ejecución por lo que puede resultar muy beneficioso o muy perjudicial, dependiendo los valores de sus parámetros y lo que haga el método.

Los métodos recursivos suelen ser muy pequeños, pero una vez desglosados, resulta en que hacen muchos procesos, además de que hacen muchas llamadas así mismo lo que provoca que el tiempo de ejecución del método tarde un poco más.