EjercicioRecursividad

Estructuras de Datos

Tema 1: tipos abstractos de datos y algoritmia

1º Grado en Ingeniería de la Computación
© Profesor Dr. Carlos Grima Izquierdo (www.carlosgrima.com)
URJC (www.urjc.es)

Escribe en C++, de forma recursiva, los siguientes procedimientos (no adjuntes un proyecto C++, sino que basta con poner en un Word el código de cada procedimiento, después de haberlo probado y asegurarte que funciona):

1. El algoritmo de Euclides para calcular el MCD de dos números naturales. Los parámetros del método son los dos números. El resultado es su MCD. Precondiciones: los dos operandos son naturales y además el primero es mayor que el segundo.
2. Un número real elevado a un número natural. Los parámetros del método son la base y el exponente. El resultado es la potencia. Precondiciones: el exponente es un número natural.
3. La suma de los n primeros números naturales. El parámetro del método es n. El resultado es la suma. Precondición: el parámetro es un número natural.
4. Calcular el término n-ésimo de la sucesión de Fibonacci. El parámetro del método es n, y el resultado es el término n-ésimo. Cada término de la sucesión se calcula sumando los dos anteriores, es decir: F(n)=F(n-1)+F(n-2). Los dos primeros términos son 0 y 1, es decir: F(0)=0 y F(1)=1. Ej: los primeros términos de la sucesión son: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, etc. Ej: F(10)=55.
5. Comprobar si un número natural existe en un vector de números naturales. Si se encuentra se devuelve true y, en caso contrario, false. El parámetro es el vector (un puntero a su comienzo), su tamaño, y el número natural a encontrar dentro de él. El resultado es true o false. Precondiciones: el tamaño debe ser mayor o igual que 0 (si es 0, lógicamente se devolverá false).
6. Realizar un método para calcular la multiplicación de dos números naturales, utilizando el algoritmo de "multiplicación a la rusa". Parámetros: dos números enteros mayores o iguales que 1. Retorno: el resultado de multiplicarlos. Para multiplicar "a la rusa", se hace una tabla con tres columnas. En la primera fila, se pone el primer operando en la primera columna y el segundo en la segunda. Se van calculando las sucesivas filas según lo siguiente:

    1. En la primera columna se pone el primer operando, y se va dividiendo entre dos (ignorando el resto) en cada fila
    2. En la segunda columna se pone el segundo operando, y se va multiplicando por dos

3. En la tercera columna se pone el segundo operando, siempre que la primera columna sea impar. Si no, no se pone nada.

Se siguen calculando nuevas filas hasta que en la primera columna queda un 1. El resultado es la suma de todas las terceras columnas no vacías.

Ejemplo: multiplicar 981 y 1234:

Operando A	Operando B	
981	1234	1234
490	2468	
245	4936	4936
122	9872	
61	19744	19744
30	39488	
15	78976	78976
7	157952	157952
3	315904	315904
1	631808	631808
	TOTAL:	**1210554**

Solución

Euclides:

```
int calcularMCD (int operando1, int operando2) {
	if (operando2==0)
		return(operando1);
	else
		return (calcularMCD(operando2,(operando1%operando2)));
}
```

Potenciación:

```
float elevarAPotencia (float base, int exponente) {
	if (exponente == 0) return 1;
	else return (base*elevarAPotencia(base,exponente-1));
}
```

Sumatorio:

```
int sumarHasta (int fin) {
	if (fin == 0) return 0;
	else return (fin+sumarHasta(fin-1));
}
```

Fibonacci:

```
int calcularFibonacci(int n) {
    if (n == 0 || n == 1) return(n);
    else return (calcularFibonacci(n - 1) + calcularFibonacci(n - 2));
}
```

Búsqueda:

```
bool buscar(int *vector, int n, int numeroABuscar) {
	if (n==0) return false;
	else
		if (vector[n-1] == numeroABuscar) return true;
		else return (buscar(vector, n-1, numeroABuscar));
}
```

Multiplicación a la rusa:

```
long multiplicarALaRusa(long operandoA, long operandoB) {
	if (operandoA==1) return(operandoB); // Caso trivial
	else {
		long nuevoA = operandoA / 2; // Nuevo primer operando
		long nuevoB = operandoB * 2; // Nuevo segundo operando
		// Distinguimos el caso de que A sea impar o par
		if ((operandoA%2)!=0) // A impar
			return (operandoB+multiplicarALaRusa(nuevoA, nuevoB));
		else // A par
			return (multiplicarALaRusa(nuevoA, nuevoB));
	}
}
```