

Algoritmos recursivos

1. Implementar uma função recursiva para calcular o valor de 2^N , com $N \geq 0$.
2. Implementar uma função recursiva para calcular o número de caracteres que contém uma frase, tendo em conta que os espaços em branco são caracteres a contabilizar e que a frase termina com '\0'.
3. Implementar uma função recursiva para calcular o número de dígitos que contém um número inteiro positivo N.
4. Implementar uma função recursiva para calcular o valor da sequência de Fibonacci de N, $\text{Fib}(N)$, com $N \geq 0$, tendo em conta que:
$$\text{Fib}(0) = 0$$
$$\text{Fib}(1) = 1$$
$$\text{Fib}(N) = \text{Fib}(N-1) + \text{Fib}(N-2), N > 1$$
5. Implementar uma função recursiva para calcular o máximo divisor comum de dois números inteiros positivos, usando o algoritmo de Euclides, que se enuncia da seguinte forma:
$$\text{MDC}(a, 0) = a$$
$$\text{MDC}(a, b) = \text{MDC}(b, a \% b), b > 0$$
6. Implementar uma função recursiva para determinar o número de divisores de um número inteiro positivo não nulo.
7. Implementar uma função recursiva para determinar o maior elemento de um array 1D de inteiros (usar as bibliotecas construídas antes para tratar os arrays).
8. Implementar uma função recursiva para determinar o índice/posição do maior elemento de um array 1D de inteiros (usar as bibliotecas construídas antes para tratar os arrays).
9. Implementar uma função recursiva para determinar a soma dos elementos de um array 1D de inteiros positivos (usar as bibliotecas construídas antes para tratar os arrays).
10. Implementar uma função recursiva para determinar a soma dos elementos pares de um array 1D de inteiros positivos (usar as bibliotecas construídas antes para tratar os arrays).
11. Implementar uma função recursiva para determinar o número de elementos pares de um array 1D de inteiros positivos (usar as bibliotecas construídas antes para tratar os arrays).
12. Implementar uma função recursiva para determinar o número de elementos positivos de um array 1D de inteiros (usar as bibliotecas construídas antes para tratar os arrays).

13. Implementar uma função recursiva para verificar se um dado número inteiro positivo não nulo N é primo ($N \geq 1$). Um inteiro é primo se e só se é divisível apenas por ele e por 1.

14. Implementar uma função recursiva para determinar o maior elemento par de um array 1D de inteiros positivos (usar as bibliotecas construídas antes para tratar os arrays).

15. As notas dos alunos estão guardadas no ficheiro de texto "Dados2.txt". Cada linha deste ficheiro contém a seguinte informação: número de aluno e nota final obtida (por esta ordem). Implementar um programa em C para determinar (a) a nota média obtida na disciplina e (b) a nota média obtida na disciplina apenas pelos alunos aprovados. Para tal, construir e usar as seguintes funções:

a) função para ler do ficheiro um array 1D de inteiros com as notas dos alunos.

int *lerArray1DAlunos (int *N);

b) função recursiva para calcular a soma das notas obtidas pelos alunos (usar o array).

int somaNotas (int V[], int N);

c) função recursiva para calcular a soma das notas dos alunos aprovados (usar o array).

int somaNotasAprovados (int V[], int N);

16. As notas dos alunos estão guardadas no ficheiro de texto "Dados3.txt". Cada linha deste ficheiro contém a seguinte informação: número de aluno, nota dos trabalhos práticos (TP) e nota do teste escrito (T) (por esta ordem). Implementar um programa em C para determinar o número de alunos aprovados. Para tal, construir e usar as seguintes funções:

a) função para ler do ficheiro um array 1D de inteiros com as notas finais (TP+T) dos alunos.

int *lerArray1DAlunos (int *N);

b) função recursiva para calcular o número de alunos aprovados (usar o array).

int calcularAprovados (int V[], int N);