Aula Prática 4 - Estruturas de Dados I (BCC202)

Marco Antonio M. Carvalho
Universidade Federal de Ouro Preto
Departamento de Computação

11 de junho de 2021

Instruções

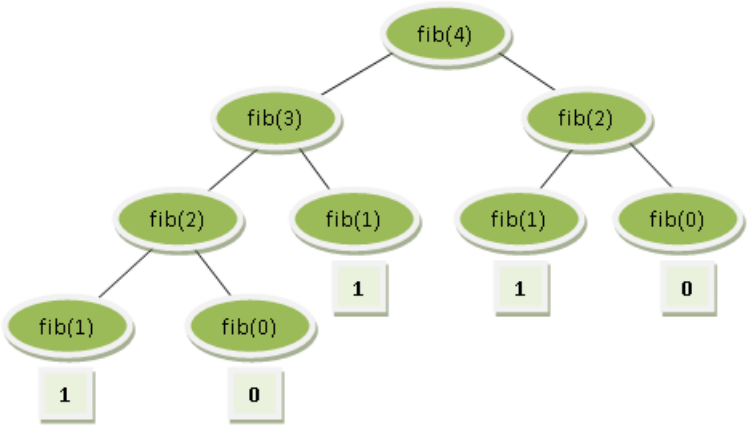
- Siga atentamente quanto ao formato da entrada e saída de seu programa, exemplificados no enunciado;
- Durante a correção, os programas serão submetidos a vários casos de testes, com características variadas;
- A avaliação considerará o tempo de execução e o percentual de respostas corretas;
- Eventualmente realizadas entrevistas sobre os estudos dirigidos para complementar a avaliação;
- Considere que os dados serão fornecidos pela entrada padrão. Não utilize abertura de arquivos pelo seu programa. Se necessário, utilize o redirecionamento de entrada;
- Os códigos fonte serão submetidos a uma ferramenta de detecção de plágios em software;
- Códigos cuja autoria não seja do aluno, com alto nível de similaridade em relação a outros trabalhos, ou que não puder ser explicado, acarretará na perda da nota e frequência;
- Códigos ou funções prontas específicos de algoritmos para solução dos problemas elencados não são aceitos;
- Não serão considerados algoritmos parcialmente implementados.

1 Recursividade

Quase todo estudante de Ciência da Computação recebe em algum momento no início de seu curso de graduação algum problema envolvendo a sequência de Fibonacci. Tal sequência tem como os dois primeiros valores 0 (zero) e 1 (um) e cada próximo valor será sempre a soma dos dois valores imediatamente anteriores. Por definição, podemos apresentar a seguinte fórmula para encontrar qualquer número da sequência de Fibonacci:

```
fib(0) = 0
fib(1) = 1
fib(n) = fib(n-1) + fib(n-2);
```

Uma das formas de encontrar o número de Fibonacci é através de chamadas recursivas. Isto é ilustrado a seguir, apresentando a árvore de derivação ao calcularmos o valor fib(4), ou seja o quinto valor desta sequência:



Desta forma,

- fib(4) = 1+0+1+1+0 = 3
- Foram feitas 8 chamadas recursivas.

Especificação da Entrada

A primeira linha da entrada contém um único inteiro n, indicando o número de casos de teste. Cada caso de teste contém um número inteiro, a partir do qual deve ser calculado o fibonacci.

Especificação da Saída

Para cada caso de teste de entrada deverá ser apresentada uma linha de saída, no seguinte formato: fib(n) = **x** chamadas = **y**, em que **x** é o número de chamadas recursivas e **y** representa o número fibonacci encontrado. Há sempre um espaço antes e depois do sinal de igualdade, conforme o exemplo abaixo.

Exemplo de Entrada

```
2
5
4
```

Exemplo de Saída

```
fib(5) = 14 chamadas = 5
fib(4) = 8 chamadas = 3
```

Estrutura do código

O código-fonte deve conter uma função recursiva para efetuar o cálculo desejado e deve ser modularizado corretamente conforme os arquivos de protótipo fornecidos. A informação de cada duende deve ser armazenada em um tipo abstrato de dados criado especificamente para isso, contendo campos *n* (para armazenar o número fornecido na entrada), *resultado* (para armazenar o número fibonacci calculado) e *chamadas* (para armazenar o número de chamadas recursivas realizadas).

Note que os dois últimos valores podem ser números muito grandes e, portanto, um tipo adequado deve ser utilizado.

Diretivas de Compilação

```
$ gcc fib.c -c
$ gcc principal.c -c
$ gcc fib.o principal.o -o programa
```