Universidade do Estado do Amazonas
Escola Superior de Tecnologia
Data: 21 de abril de 2015
Disciplina: Introdução à Programação de Computadores
Professora: Elloá B. Guedes
Aluno:

4ª Lista de Exercícios
Recursividade

1. Implemente um algoritmo recursivo em Python que calcule o resto da divisão de dois números inteiros x e y utilizando a seguinte definição:

$$mod(x,y) = \begin{cases} mod(|x|-|y|,|y|, & \text{se } |x| > |y| \\ |x|, & \text{se } |x| < |y| \\ 0, & \text{se } |x| == |y| \end{cases} \quad (1)$$

   Se não for possível efetuar o cálculo, faça com o que sua função retorne -1. Para testar sua função, construa asserts comparando o resultado obtido com o do operador % em Python.

2. Pode-se calcular o quociente da divisão de x por y, dois números inteiros, utilizando-se a seguinte definição:

$$div(x,y) = \begin{cases} 1 + div(|x|-|y|,|y|, & \text{se } |x| > |y| \\ 0, & \text{se } |x| < |y| \\ 1, & \text{se } |x| == |y| \end{cases} \quad (2)$$

   Implemente esta função em Python utilizando a definição recursiva apresentada. Para os casos em que não for possível efetuar este cálculo, retorne -1.

3. Escreva um algoritmo recursivo para implementar a exponenciação, utilizando a seguinte estratégia de resolução:

$$\begin{cases} base^{exp} = base * base^{exp-1}, & \text{se } exp > 0 \\ base^{exp} = 1, & \text{se } exp = 0. \end{cases} \quad (3)$$

4. Faça um algoritmo em Python utilizando a seguinte estratégia recursiva para implementar a exponenciação:

$$\begin{cases} base^{exp} = 1, & \text{se } exp = 0. \\ base^{exp} = (base^2)^{\frac{exp}{2}}, & \text{se } exp \text{ é par}, \\ base^{exp} = base * base^{exp-1}, & \text{se } exp \text{ é ímpar}. \end{cases} \quad (4)$$

5. Quais as diferenças e semelhanças nos algoritmos recursivos para a exponenciação? Qual deles faz menos chamadas recursivas?

6. O máximo divisor comum de dois números inteiros positivos é o maior inteiro que divide ambos sem deixar resto. Por exemplo, $gcd(2, 12) = 2$, $gcd(6, 12) = 6$, $gcd(9, 12) = 3$, $gcd(17, 12) = 1$. O algoritmo de Euclides para encontrar o máximo divisor comum entre dois inteiros a e b é um algoritmo recursivo, que funciona da seguinte forma:

   (a) Se b é zero, a resposta é a;

   (b) Senão, $gcd(a, b)$ é o mesmo que $gcd(b, a\%b)$.

   Escreva um algoritmo recursivo em Python para encontrar o máximo divisor comum entre dois números utilizando a estratégia criada por Euclides. Faça também o algoritmo iterativo equivalente.

7. Escreva uma função recursiva que escreva na tela todos os números inteiros positivos desde um valor k informado pelo usuário até 0.

8. Escreva um algoritmo recursivo que escreva na tela a soma de todos os números inteiros positivos de k até 0.

9. Escreva uma função recursiva que calcule a soma de todos os números compreendidos entre os valores a e b passados por parâmetro, em que $a < b$.

10. Escreva uma função recursiva que implemente o equivalente a função `len` para strings. Isto é, sua função deve receber como entrada uma string e retornar o número de caracteres que esta string possui. Considere o caso base como sendo o da string vazia, cujo comprimento é zero.

11. Escreva um programa em Python que implemente um algoritmo recursivo para o Fibonacci.

12. Escreva um programa recursivo em Python que receba um número inteiro e positivo n e que retorne um triângulo de asteriscos com n linhas, em que o número de asteriscos é igual ao número correspondente da linha. Veja o exemplo a seguir considerando $n = 5$.

```
*
**
***
****
*****
```

13. Modifique o programa anterior e, ainda utilizando uma estratégia recursiva, imprima o triângulo invertido. Veja o exemplo para $n = 5$.

```
*****
****
***
**
*
```

14. Escreva uma função recursiva que calcule os juros compostos de um valor. Para isso o programa deverá ler um valor inicial, o número de meses e a taxa de juros ao mês, e passar estes valores à função como parâmetros.

15. Faça uma função recursiva que permita inverter um número inteiro. Por exemplo, `inverte(123) = 321`.

16. O superfatorial de um número n é definida pelo produto dos n primeiros fatoriais de n. Assim, o superfatorial de 4 é $sf(4) = 1!*2!*3!*4! = 288$. Faça uma função recursiva que receba um número inteiro positivo n e retorne o superfatorial desse número. Utilize a função fatorial recursiva implementada anteriormente.

17. Uma palavra de Fibonacci é definida por:

$$f(n) = \begin{cases} b, & \text{se } n = 0 \\ a, & \text{se } n = 1 \\ f(n-1) + f(n-2), & \text{se } n > 1 \end{cases} \tag{5}$$

em que o símbolo + denota a concatenação de duas strings. Esta sequência inicia com as seguintes palavras: b, a, ab, aba, $abaab$, $abaababa$, $abaababaabaab$, ... Faça uma função recursiva que receba um número n e retorne a n-ésima palavra de Fibonacci.

18. (DESAFIO) A McDonald's vence McNuggets em pacotes de 6, 9 ou 20 McNuggets. Assim, é possível, por exemplo, comprar exatamente 15 McNuggets (um pacote de 6 mais um pacote de 9), mas não é possível comprar exatamente 16 McNuggets, uma vez que não há uma combinação dos pacotes disponíveis que somem 16. Para determinar se é possível comprar exatamente n McNuggets, alguém precisa encontrar um valor inteiro não negativo (pode ser nulo) de a,b e c tal que $6 \cdot a + 9 \cdot b + 20 \cdot c = n$. Escreva uma função que recebe um argumento n e que retorna True caso seja possível comprar n McNuggets por meio de uma combinação dos pacotes com 6, 9 e 20 McNuggets que seja exatamente igual a n. Em caso contrário, retorne Falso. Utilize um algoritmo recursivo para resolver este problema.