Aula 12: Recursão

Professor(a): João Eduardo Montandon
Virgínia Fernandes Mota
jemaf.github.io
http://www.dcc.ufmg.br/∼virginiaferm

INTRODUÇÃO A PROGRAMAÇÃO - SETOR DE INFORMÁTICA

Recursividade: O que é??

Um objeto é **recursivo** quando é **definido** parcialmente **em termos de si mesmo**.

Entendendo a recursividade

Suponha que alguém no ICEx lhe pergunte:
"Como chego à Rua Monteiro Lobato?"

Entendendo a recursividade

"Como chego à Rua Monteiro Lobato?"

- Você poderia responder com conjunto completo de direções e referências... mas, se as direções são muito complexas, ou se você não está muito certo de como chegar lá, poderia responder:
- **"Vá até a saída da Catalão e então pergunte lá: "Como chego à Rua Monteiro Lobato?"**
- O clássico: Segue toda a vida e pergunte de novo lá na frente.

Este é um exemplo de recursão

Revendo o problema...

- Seu colega queria direções para um destino.
- Para resolver o problema, você deu um conjunto de instruções indicando como seu colega poderia alcançar seu objetivo.
- Após chegar ao destino indicado por você, seu colega se depara com nova versão do problema original.

Novo problema, idêntico em forma ao original, envolve uma nova localização de partida mais próxima ao destino final!

Objetos recursivos – Exemplo I

Números Naturais

- **Caso base:** 1 é um número natural
- **Passo recursivo:** o sucessor de um número natural é um número natural

Objetos recursivos – Exemplo II

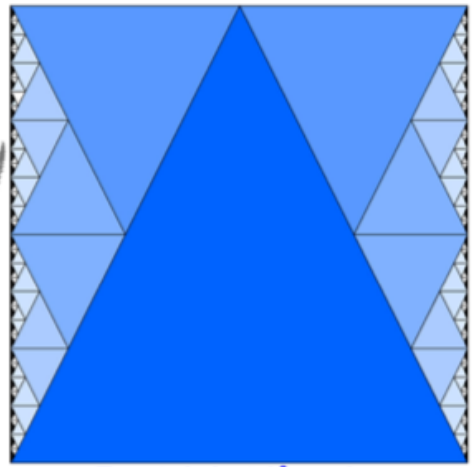
Fatorial ($n!$)

- **Caso base:** $0! = 1$
- **Passo recursivo:** se $n > 0$ então $n! = n(n-1)!$

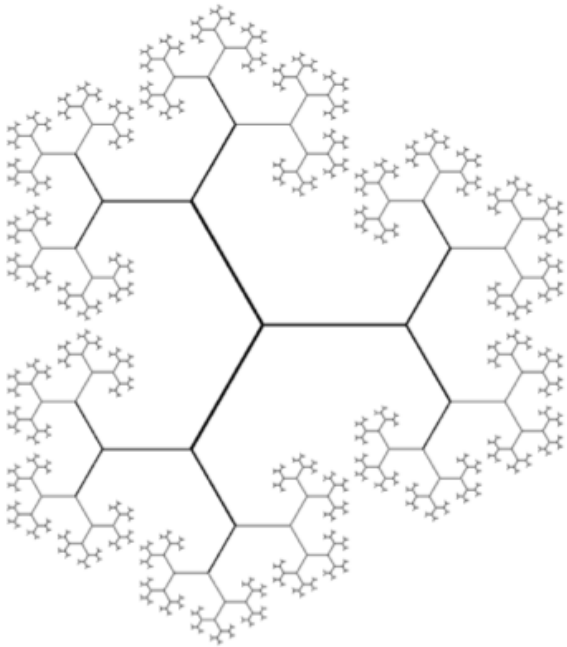
Objetos recursivos

Fractal de Fern

Fractal espiral



Fractal de triângulos



Derivação binária

https://www.youtube.com/watch?v=58dmCj0wuKw

Objetos recursivos

Foto recursiva

Imagem recursiva

Pensamento recursivo

Definição

- Em programação, uma função é dita **recursiva** quando chama a si própria, direta ou indiretamente.
- Propósito: expressar sua funcionalidade ou objetivo de maneira mais clara e concisa.
- Linguagens que permitem uma função chamar a si própria são ditas recursivas.
    - Linguagens que não permitem são ditas iterativas ou não recursivas

Exemplo de Algoritmo Iterativo

- Cálculo do fatorial de um número **n**.
- Sabemos que n! = n*(n-1)*(n-2)*...*(1).
- Implementação **iterativa**:

```
1 int fatorial(int n){
2     int fat = 1, i;
3     for (i = 2; i <= n; i++)
4         fat = fat*i;
5     return fat;
6 }
```

Exemplo de Algoritmo Recursivo

- Tanto a definição matemática quanto o código anterior são simples, porém mais elegante definir o fatorial como:

$$n! = \begin{cases} 1, & se\ n = 1 \\ n(n-1)!, & se\ n > 1 \end{cases} \quad (1)$$

- Fatorial de **n** definido a partir dos fatoriais dos naturais menores que ele.
- Significa que para cálculo do fatorial de um determinado número há necessidade de que se recorra aos fatoriais dos números anteriores.

Exemplo de Algoritmo Recursivo

- Condição que define parada da recursão: chamada **base da recursão** ou de **caso base**.
- Todo programa recursivo deve possuir uma **condição de parada**!
- Condição que define recursão chamada **passo recursivo**.

Exemplo de Algoritmo Recursivo

Definição matemática:

$$n! = \begin{cases} 1, & se\ n = 1 \\ n(n-1)!, & se\ n > 1 \end{cases} \quad (2)$$

Algoritmo Recursivo:

```
1 int fatorial(int n){
2     if(n == 1)
3         return 1; //Base da recursao
4     else
5         return (n*fatorial(n-1)); //Passo recursivo
6 }
```

Exemplo

Escreva um código recursivo para calcular o máximo divisor comum, MDC, de dois números inteiros positivos N e M da seguinte maneira:

$$mdc(n, m) = \begin{cases} m, & se\ n \geq m\ e\ n\ mod\ m = 0 \\ mdc(m, n), & se\ n < m \\ mdc(m, n\ mod\ m), & caso\ contrario \end{cases}$$

Exemplo

```c
#include <stdio.h>
int mdc(int n, int m){
    if ( (n >= m) && ((n % m)==0))
        return(m);
    else {
        if (n < m)
            return (mdc(m,n));
        else
            return (mdc(m,n %m));
    }
}

int main(){
    int m, n, MDC;
    scanf("%d %d", &n, &m);
    MDC = mdc(n, m);
    return 0;
}
```

$$mdc(n,m) = \begin{cases} m, & se\ n \geq m\ e\ n\ mod\ m = 0 \\ mdc(m,n), & se\ n < m \\ mdc(m, n\ mod\ m), & caso\ contrario \end{cases}$$

Exemplo

Escreva um código recursivo diferente para o MDC

Exemplo

```c
#include <stdio.h>
int mdc(int n, int m){
    if (m == 0)
        return n;
    else
        return mdc(m, n % m);
}

int main(){
    int m, n, MDC;
    scanf("%d %d", &n, &m);
    if (n > m)
        MDC = mdc(n,m);
    else
        MDC = mdc(m,n);
    return 0;
}
```

Recursividade: Aplicações

A recursividade utilizada extensivamente em vários nichos da computação:

- Computação Gráfica
- Algoritmos Genéticos
- Ordenação de elementos
- Estruturação e Busca de informações
- Algoritmos de Compactação
- etc.. etc..

Exercícios

1. Faça um procedimento recursivo que receba dois valores inteiros a e b e imprima o intervalo fechado entre eles. Se a for maior que b mostrar mensagem de erro.
2. Seja S um vetor de inteiros. Descreva funções recursivas para calcular:
    1. o elemento máximo de S;
    2. a soma dos elementos de S;
    3. média aritmética dos elementos de S.
3. Escreva uma função recursiva para calcular o N-ésimo número da sequência de Fibonacci.
4. Escreva uma função recursiva para determinar o número de dígitos de um inteiro N.
5. Escreva uma função recursiva que busque por um determinado elemento em um vetor de inteiros.

Desafio: Busca Binária

A busca binária é uma forma extremamente eficiente de buscar determinados elementos em um vetor. Para que a busca funcione corretamente, os elementos do vetor devem ser ordenados previamente.

1. Pesquise a respeito da busca binária: na Internet: Por que ela é tão rápida?? O que a difere de uma busca normal??
2. Implemente uma função recursiva que realize a busca binária em um vetor de números inteiros.

Na próxima aula...

Arquivos