Fundamentos de programação

Recursividade

Edson Moreno

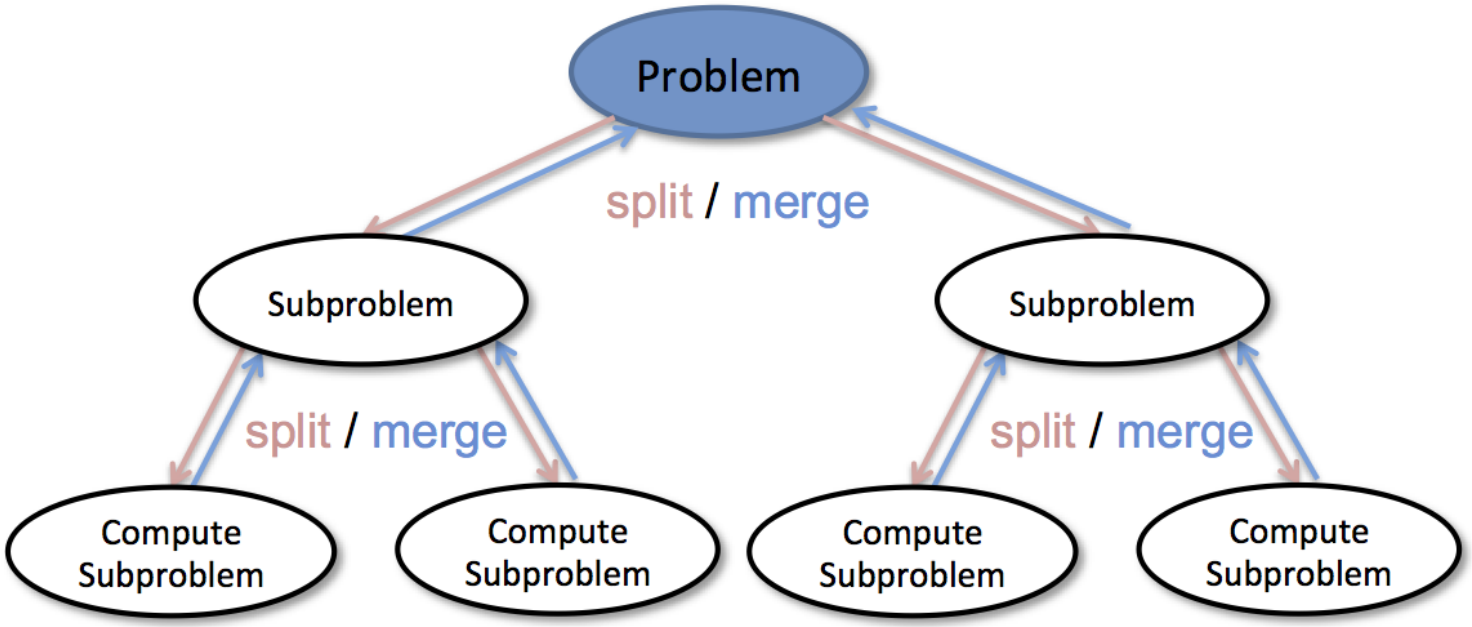
edson.moreno@pucrs.br

http://www.inf.pucrs.br/~emoreno

O que é recursividade?

- **Técnica baseada em "divisão e conquista"**
  - Buscar solucionar o problema dividindo este em problemas menores

- **Na prática em programação**
  - É a possibilidade que um método tem de invocar a si mesmo



Porque usar recursividade?

- **Muitos problemas têm a seguinte propriedade**
  - Cada instância do problema contém uma instância menor do mesmo problema.

- **Uma computação recursiva permite**
  - Resolve um problema usando a mesma solução
  - Redimensionar n vezes o problema para torna-lo mais simples
  - Resolver o problema mais simples e escalar para o original

Exemplo de funcionamento

- **Triângulo recursivo**
  - Método invoca ele mesmo
    - O parâmetro recebido é repassado, com valor decrementado
      - Objetivo é simplificar o problema original
  - Enquanto o parâmetro recebido for maior que 1
    - Nada é retornado/impresso
  - Considere a chamada do método, passando originalmente o valor 3

```
public static void printTriangle(int sideLength) {
  if (sideLength < 1) { return; }

  printTriangle(sideLength - 1);
  for (int i = 0; i < sideLength; i++) {
    System.out.print("[]");
  }
  System.out.println();
}
```

Exemplo de funcionamento

- **Triângulo recursivo**
  - Execução seria como segue
    - A chamada printTriangle(3) chama printTriangle(2)
    - A chamada printTriangle(2) chama printTriangle(1)
    - A chamada printTriangle(1) chama printTriangle(0)
    - A chamada printTriangle(0) chama retorna, sem fazer nada
    - A chamada printTriangle(1) imprime []
    - A chamada printTriangle(2) imprime [][]
    - A chamada printTriangle(3) imprime [][][]

```
public static void printTriangle(int sideLength) {
  if (sideLength < 1) { return; }

  printTriangle(sideLength - 1);
  for (int i = 0; i < sideLength; i++) {
    System.out.print("[]");
  }
  System.out.println();
}
```

Exemplo de funcionamento

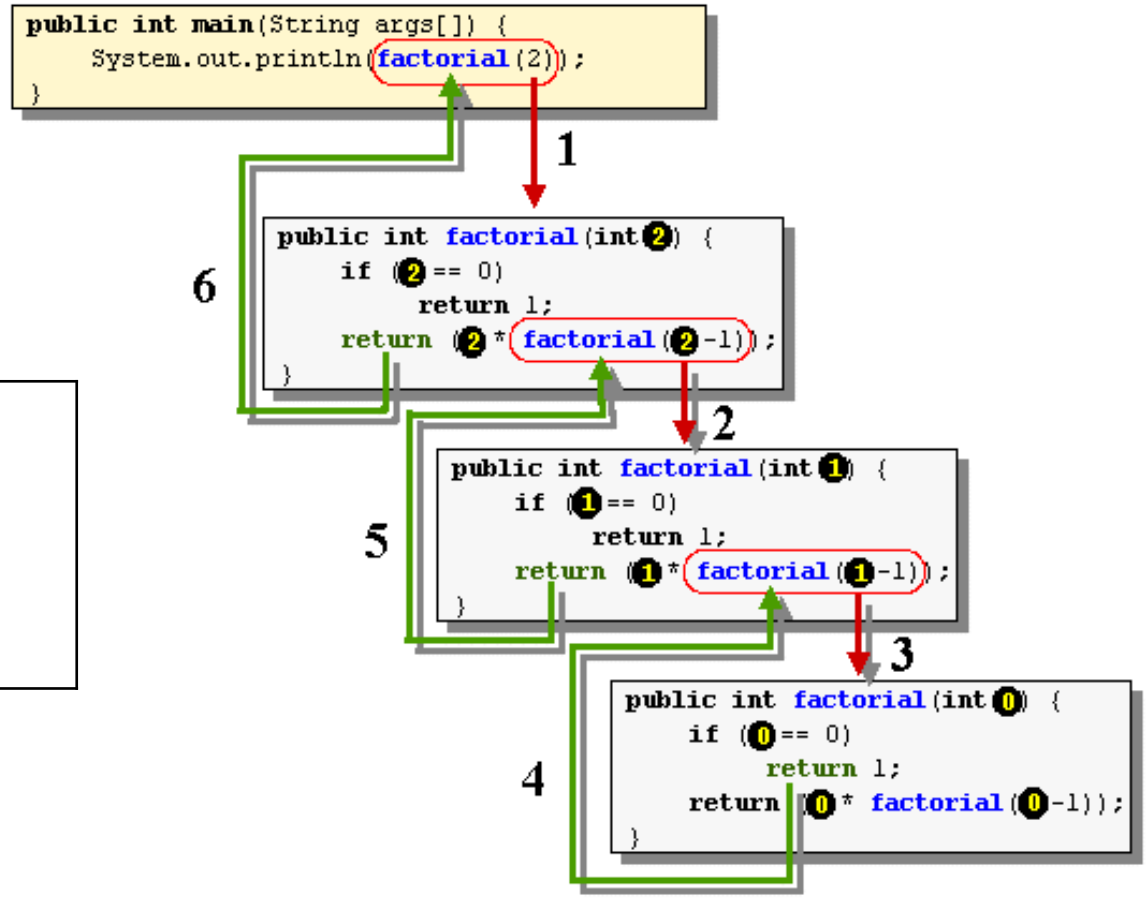
- **Fatorial recursivo**
  - Considere o método fatorial

```
public static int factorial(int n) {
   if (n == 1)  // Menor versão do problema
      return 1;
   return n * factorial(n - 1);   // passos de redução do problema
 }
```

Exemplo de funcionamento

- **Fatorial recursivo**
  - Observe o seguinte exemplo de execução

```
public static int factorial(int n) {
   if (n == 1)
      return 1;
   return n * factorial(n - 1);
}
```



Pensando em métodos recursivos

- **Decida o nome do métodos e quais parâmetros**
  - Equivalente ao que é feito para métodos convencionais.
- **Entenda se a solução recursiva é possível**
  - Se é possível resolver por iteração, é possível com recursão
  - Pense na solução para o caso mais simples do problema
    - Qual a solução quando o problema tem o menor tamanho
  - Pense em como quebrar o problema em tamanhos menores
    - Mantenha o formato do problema original
- **Determine uma técnica para quebrar o problema**
  - Alguns podem ser feitos com parâmetros simples (importante)

Pensando em recursivamente

```
Se ( o problema é suficientemente pequeno ) então
        resolva o problema
        retorne a solução
Senão
        quebre o problema em um problema menor
                Mantenha o mesmo formato do problema original
        encaminhe este problema menor para solução
        combine o resultado
        retorne a solução
Fim se
```

Tenha em mente

- **Condição de fim**
  - A não definição correta de uma condição de fim pode acarretar em recursão infinita ou erro na solução esperada
- **Funções de recursividade mútua**
  - Funções recursivas podem chamar umas as outras, criando o conceito de funções recursivas mútuas,
    - Usado em vários problemas, tal como parsers e compiladores
- **Uso de recursos em funções recursivas**
  - O custo computacional de tempo de execução e espaço de uso de memória de funções recursivas pode ser muito alto
  - Sempre considere se soluções iterarivas (for/while) podem ser mais adequadas.

Exercícios

- Qual o valor de X (4) se X é dada pelo seguinte código?

```
public static int recursivo (int n) {
  if (n == 1 || n == 2) return n;
  else return recursivo (n-1) + n * X (n-2);
}
```

- Qual o valor de ff(7) se ff é dado pelo seguinte código?

```
public static int ff (int n) {
  if (n == 1) return 1;
  if (n % 2 == 0) return ff (n/2);
  return ff ((n-1)/2) + ff ((n+1)/2);
}
```

Exercícios

- Crie um programa baseado em recursão que solicita um número positivo maior do que zero e realize o somatório dos valores que vão de 1 até o número informado.

- Escreva uma função recursiva que receba um inteiro positivo n e devolva a soma dos dígitos decimais de n. Por exemplo, ao receber 1729 sua função deve devolver 19.

- Escreva uma função recursiva que receba inteiros positivos k e n e calcule k^n.

Exercícios

- A função de Fibonacci é definida assim:
  - F(0) = 0
  - F(1) = 1
  - F(n) = F(n-1) + F(n-2) para n > 1.
- Descreva a função F de forma recursiva. Considere a figura abaixo para auxiliar na sua elaboração da solução.