Laboratório de Programação II

Recursividade

Universidade Federal de Juiz de Fora
Departamento de Ciência da Computação

Aula de Hoje

- Objetivo: implementar algoritmos recursivos.
- Revisão de algoritmos recursivos.
- Exercícios.

Recursividade

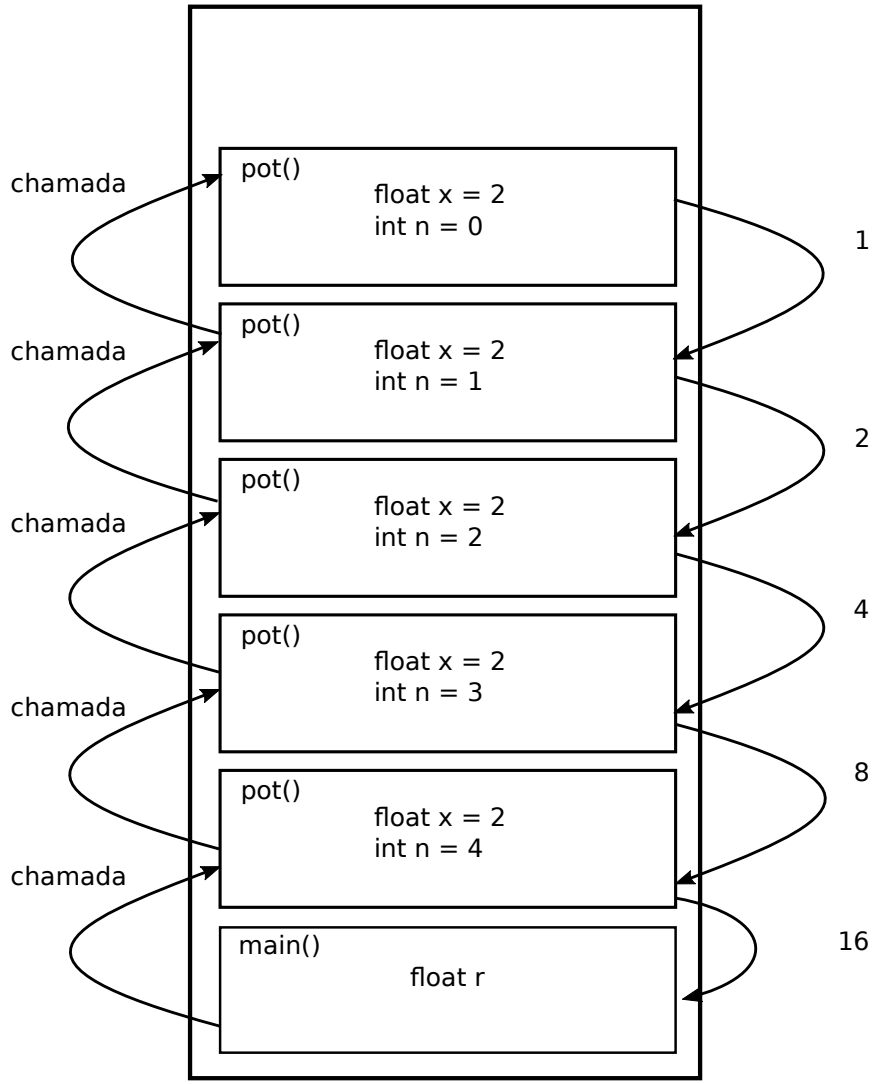
- Exemplo: função para calcular x^n.

```
float pot(float x, int n)
{
  if(n==0)
    return 1.0;
  else if(n<0)
    return pot(1.0/x, -n);
  else
    return x * pot(x, n-1);
}
```

Recursividade

```
float pot(float x, int n)
{
  if(n==0)
    return 1.0;
  else if(n<0)
    return pot(1.0/x, -n);
  else
    return x * pot(x, n-1);
}
```

```
int main()
{
  float r = pot(2,4);
  cout << r;
  return 0;
}
```



Pilha de execução

Exercícios

1. (Aquecimento) Implemente uma função **recursiva** para calcular o fatorial de um número inteiro `n`.

```
int fatorial(int n);
```

2. Desenvolva uma função **recursiva** que, dados três valores inteiros como parâmetros `a`, `b` e `inc`, imprime o intervalo fechado de `a` até `b` com incremento `inc`.

```
void imprimeIntervalo(int a, int b, int inc);
```

   Exemplo: `imprimeIntervalo(1,8,2)`
   Saída: `1 3 5 7`

3. Desenvolver uma função **recursiva** que recebe um valor inteiro `n` e imprime todos os inteiros de `n` até 0 de forma decrescente.

```
void imprimeDecrescente(int n);
```

Exercícios

4. Desenvolver uma função recursiva que recebe um vetor de reais e seu tamanho `n`, calcular e retornar a soma de todos os seus valores.

```
float soma(int n, float vet[]);
```

5. Desenvolver uma função **recursiva** que recebe um vetor de números reais e o seu tamanho `n`, calcula e retorna o menor valor do vetor.

```
float menor(int n, float vet[]);
```

6. Desenvolver uma função recursiva para calcular e retornar a quantidade de valores pares de um vetor com `n` números inteiros.

```
int contaPares(int n, int vet[]);
```

Exercícios

7. Agora, resolva todos os exercícios anteriores de forma **iterativa**, isto é, sem usar recursividade e, em seguida, compare a solucão em cada caso.