Tecnologia em Sistemas para Internet

# DISCIPLINA:
# ESTRUTURAS DE DADOS

## RECURSÃO
## ALOCAÇÃO DINÂMICA DE MEMÓRIA


- Prof. Alexandre Ponce de Oliveira

alexandreponceo@gmail.com


Aulas 23 e 24

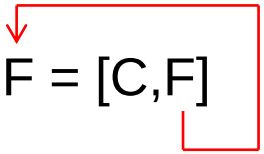
Lins/SP

# Recursão

Um algoritmo que resolve um problema dividindo-o em problemas menores, cujas soluções requerem a aplicação dele mesmo é chamado recursivo.

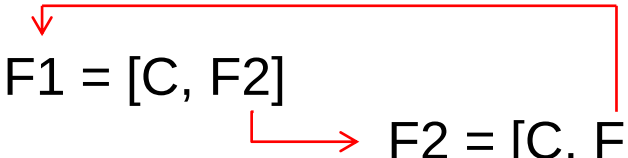
Uma função é recursiva quando dentro do seu código existe uma chamada para si própria.

Por exemplo:




Direta


Indireta




Mútua

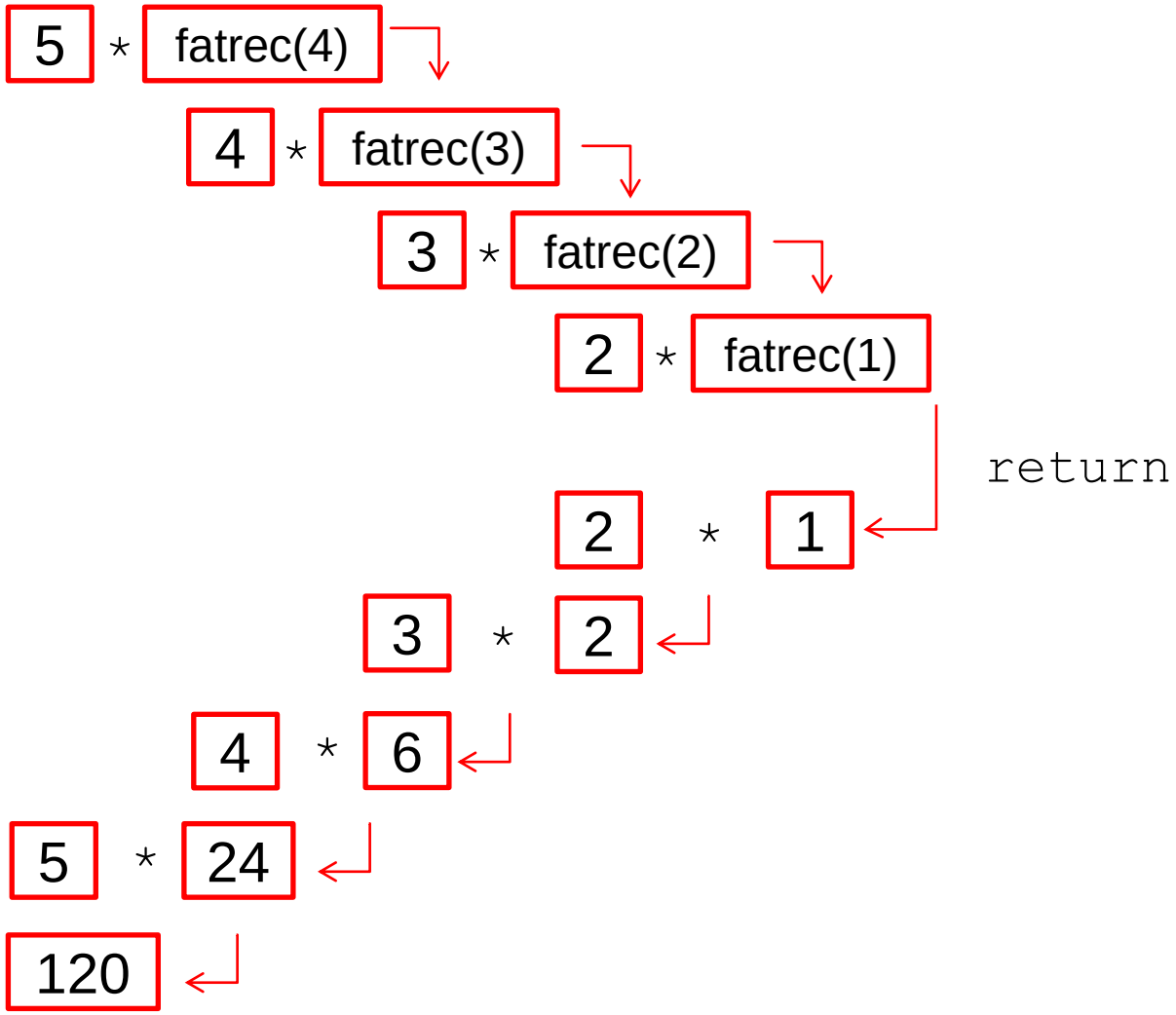
# Recursão

Exemplo: fatorial de um número.

```
//43340 notebook travou
#include <stdio.h>
#include <conio.h>
int fatrec(int n)
{
    //printf("\nChamando fatorial de %d: ", n);
    if (n >= 0 && n <= 1)
      return 1;
    return n * fatrec(n-1);
}
int main()
{
    int n;    // numero dado
    printf("Digite o valor de n: ");
    scanf("%d", &n);
    printf("\n\nfatorial de %5d = %5d", n, fatrec(n));
    getch();
}
```

# Recursão

Exemplo: fatorial do número 5.



# Recursão

## Exemplo: números pares ou ímpares.

```
//32382 travou
#include <stdio.h>
#include <conio.h>

int par(int n);

int impar(int n);

int par(int n)
{
    int par;
    par=n;
       if (par == 0)
       printf("Par");
       else
          impar(par-1);
}
```

```
int impar(int n)
{
    int impar;
    impar=n;
       if (impar == 0)
       printf("Impar");
       else
        par(impar-1);
}
int main()
{
    int n;    // numero dado
    printf("Digite um numero:
");
    scanf("%d", &n);
    par(n);
    getch();
}
```

# Alocação de Memória

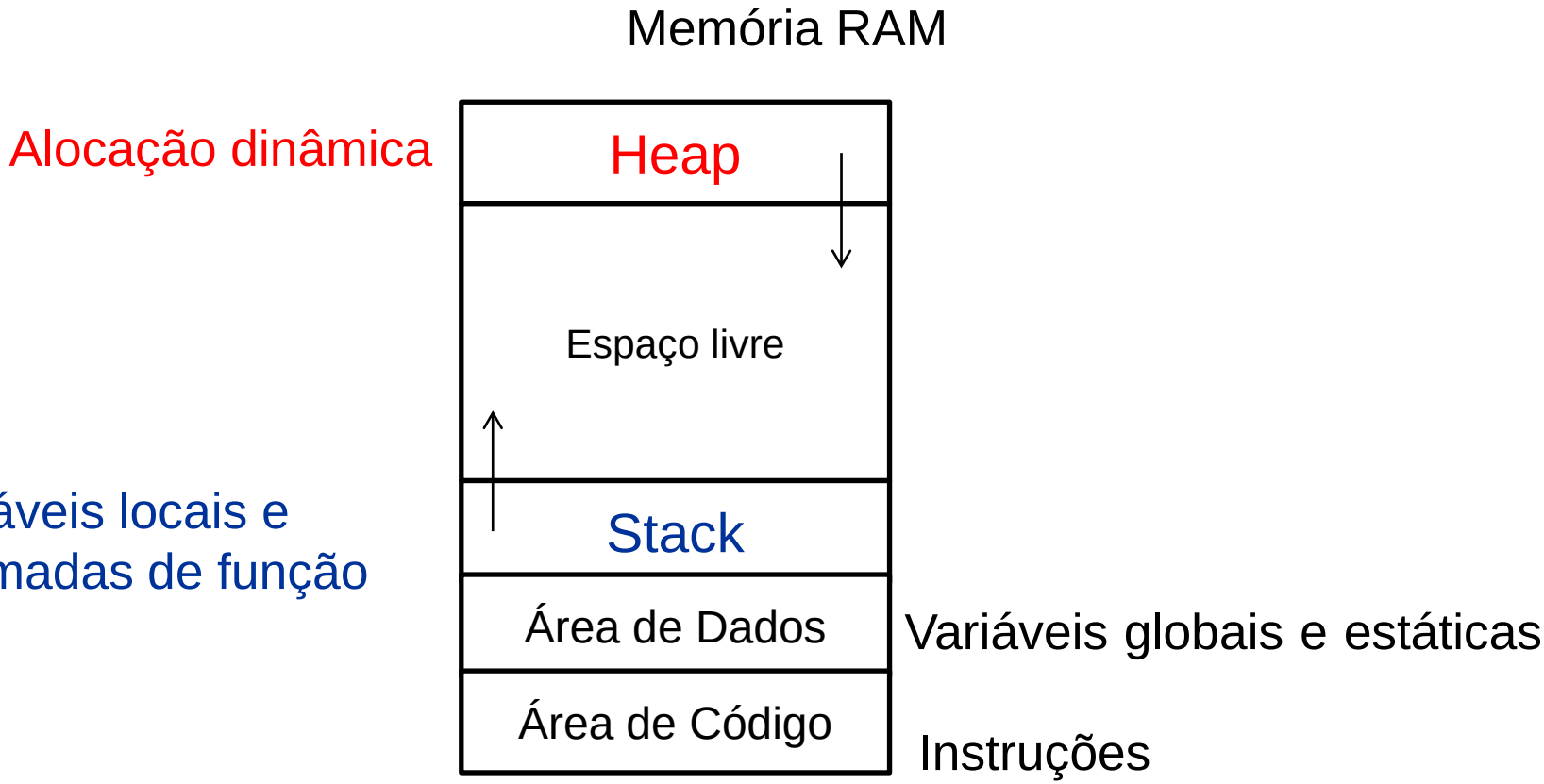
Alocação estática ocorre em tempo de compilação, ou seja, quando se define uma variável ou estrutura é necessário que se definam seu tipo e tamanho.

```
char n;
int x;
int v[5];
```

Alocação dinâmica ocorre em tempo de execução. As variáveis e estruturas são declaradas sem a necessidade de especificar seu tamanho, pois a memória será reservada apenas quando forem utilizadas.

# Alocação de Memória

Cada programa utiliza a memória da seguinte forma:



# Alocação Dinâmica de Memória

Linguagem C fornece as seguintes funções para alocar memória dinamicamente:

malloc

calloc

free

realloc

# Função malloc

```
void * malloc(int qty_bytes_alloc);
```

Solicita alocação dinâmica de memória.

É necessário informar a quantidade de bytes para alocação.

A função retorna um ponteiro para o endereço de memória alocada.

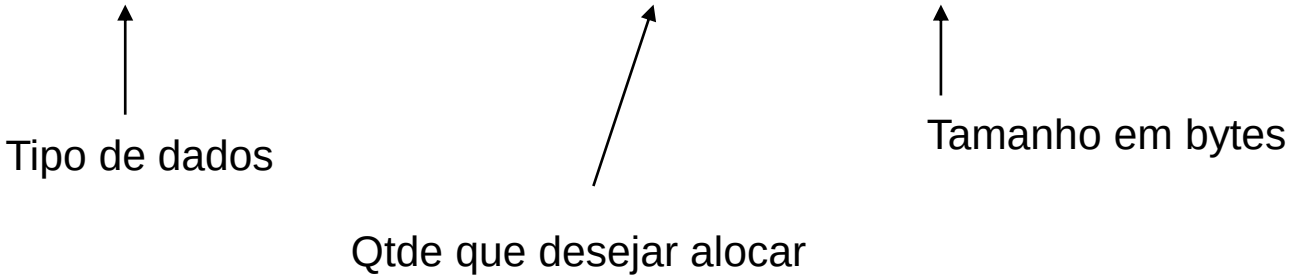
Se ocorrer erro na alocação, a função retorna NULL.

# Função malloc

Exemplo:

Alocação dinâmica para um vetor de 15 inteiros.

```
int *v;
v = (int *) malloc(15 * sizeof(int));
```



# Função calloc

void * calloc(int qty, int size);

Idem a função malloc, a diferença é que inicializa o espaço alocado com zero.

Exemplo:

Alocação dinâmica para um vetor de 15 floats.

```
float *v2, *v3;
v2 = calloc(15, sizeof(float));
//ou
v3 = (float *) calloc(15, sizeof(float));
```

# Função realloc

void * realloc(void * pointer, int new_size);

Aumenta ou diminui dinamicamente uma área de memória previamente alocada.

Recebe o ponteiro para o vetor e retorna o novo espaço alocado.

Exemplo:

```
int *v4;
v4 = (int *) calloc(15, sizeof(int));
float *v5;
v5 = calloc(15, sizeof(float));
v4 = (int *)(realloc(v4,(15+5)*sizeof(int)));
v5 = (float *)(realloc(v5,(15-5)*sizeof(float)));
```

# Função free

free( void * pointer);

Libera uma área de memória alocada pelas funções malloc, calloc ou realloc.

Recebe como parâmetro o ponteiro para a área de memória alocada.

Exemplo:

```
int *v;
v = (int *) malloc(15 * sizeof(int));
free(v);
```

# Exemplo – Alocação Dinâmica de Vetor

```
#include <stdio.h>
#include <conio.h>
main() {
  int *v;
  int i,j;
  printf("\nDigite o tamanho do vetor: ");
  scanf("%d", &i);
  v = (int *)(malloc(i*sizeof(int)));
  if( v == NULL )
       printf("\nErro para alocacao de memoria");
  else       {
       for( j=0;j<i;j++)          {
              printf("Digite os numeros do vetor: ");
              scanf("%d", &v[j]);        }
       printf("Os numeros digitados foram: ");
       for( j=0;j<i;j++)
              printf("%d ", v[j]);
       free(v);
  }
  getch();
}
```

# Exemplo – Teste de Memória

```
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
int main() {
  char *w;
  int i, n=100;
  w = (char *)(malloc(100*sizeof(char)));
  if( w == NULL )
       printf("\nErro para alocacao de memoria");
  for (i = 100; ; i=i*2)
  {
     w = (char *)(realloc(w,i*sizeof(char)));
     if( w == NULL ) {
         printf("\nErro para alocacao de memoria");
         getch();
         exit(0);
       }
     else
        printf( "Alocado %d \n",i);
  }
  free(w);
}
```

# Exercícios

1) Escreva um programa para alocar dinamicamente memória para um vetor definido pelo usuário (utilize a função calloc). Preencha e mostra os números do vetor.

2) Complemente o programa anterior aumentando o tamanho do vetor de acordo com o tamanho definido pelo usuário. Adicione mais números no vetor aumentado e mostre todos os números.