

Linguagens de Programação 1

**Francisco Sant'Anna
Sala 6020-B
francisco@ime.uerj.br**

<http://github.com/fsantanna-uerj/LP1>

Alocação Dinâmica

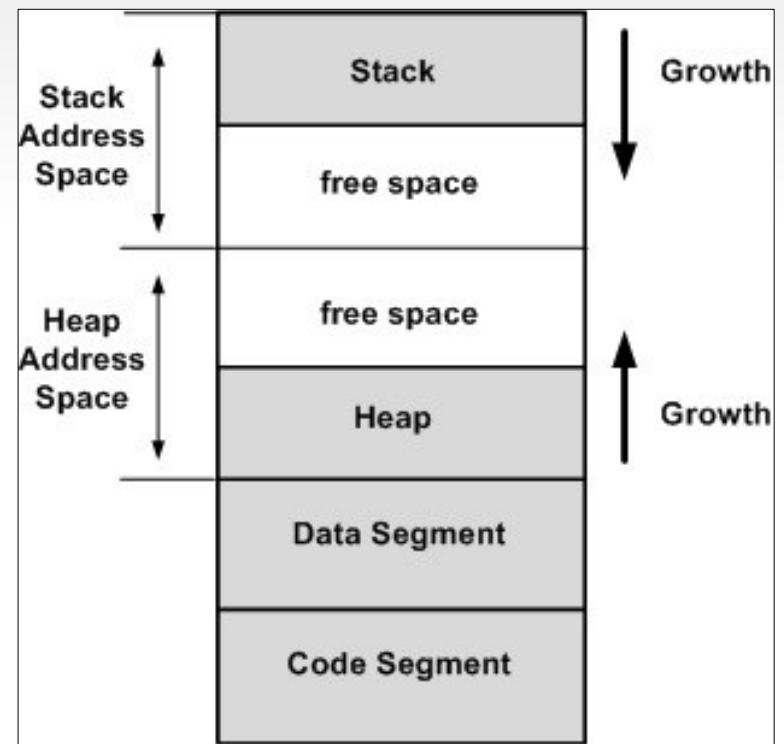
Alocação de Memória

```
static int v[5];

void f (int i, int x) {
    if (i == 5) {
        return;
    }

    v[i] = x;
    f(i+1, x*2);
    printf("%d %d\n", i, x);
}

int main (void) {
    f(0, 10);
    return 0;
}
```



Exercício 9.1 (revisão)

- Criar um vetor `vet` de 5 posições
- Ler 5 números e guardá-los em `vet`
- Exibir todos os números de `vet`
- Ler um outro número `I`
- Remover o valor de `vet` no índice `I`
 - Manter o vetor sem buracos

Exercício 9.1 - Problemas?

- Precisamos re-ajustar as posições o tempo todo.
- E se eu precisar de mais um elemento?

Alocação Dinâmica

- Necessário quando é impossível prever o uso total de não locais antes de executar o programa.
 - `malloc`: aloca um bloco de memória na heap
 - `free`: desaloca o bloco de memória

```
typedef struct Caixa {  
    ...  
} Caixa;  
  
int main (void) {  
    ...  
    Caixa* p = (Caixa*) malloc(sizeof(Caixa));  
    ...  
    free(p);  
    ...  
}
```

malloc

- `void* malloc (size_t size)`
 - (?) `void*`: exige typecast
 - (?) `size_t`: garante alcance
- `int* ptr = (int*) malloc(30*sizeof(int))`
 - `(int*)`
 - `sizeof`