LABORATÓRIO DE ALGORITMOS E ESTRUTURAS DE DADOS I

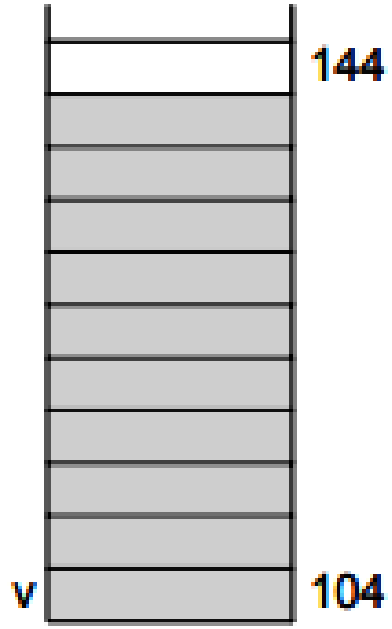
Prof. Caio César de Freitas Dantas

Ponteiros e Alocação Dinâmica

int v[10];

A declaração acima diz que v é um vetor de inteiros dimensionado com 10 elementos, isto é, reservamos um espaço de memória contínuo para armazenar 10 valores inteiros

v[10] → está ERRADO (invasão de memória)



Ponteiros e Alocação Dinâmica

```c
int main ( void ) {
float v[10]; /* declara vetor com 10 elementos */
float med, var; /* variáveis para armazenar a média e a variância */
int i; /* variável usada como índice do vetor */

/* leitura dos valores */
for (i = 0; i < 10; i++) /* faz índice variar de 0 a 9 */
scanf("%f", &v[i]); /* lê cada elemento do vetor */

/* cálculo da média */
med = 0.0; /* inicializa média com zero */
for (i = 0; i < 10; i++) med = med + v[i]; /* acumula soma dos elementos */
med = med / 10; /* calcula a média */
```

Ponteiros e Alocação Dinâmica

```c
int main ( void ) {
float v[10]; /* declara vetor com 10 elementos */
float med, var; /* variáveis para armazenar a média e a variância */
int i; /* variável usada como índice do vetor */

/* leitura dos valores */
for (i = 0; i < 10; i++) /* faz índice variar de 0 a 9 */
scanf("%f", &v[i]); /* lê cada elemento do vetor */

/* cálculo da média */
med = 0.0; /* inicializa média com zero */
for (i = 0; i < 10; i++) med = med + v[i]; /* acumula soma dos elementos */
med = med / 10; /* calcula a média */
```

Ponteiros e Alocação Dinâmica

Existe uma associação forte entre vetores e ponteiros, pois se existe a declaração:

```
int v[10];
```

A variável v, que representa o vetor, é uma constante que armazena o endereço inicial do vetor, isto é, v, sem indexação, aponta para o primeiro elemento do vetor.

Ponteiros e Alocação Dinâmica

- Passar um vetor para uma função consiste em passar o endereço da primeira posição do vetor.

- Se passarmos um valor de endereço, a função chamada deve ter um parâmetro do tipo ponteiro para armazenar este valor.

- Assim, se passarmos para uma função um vetor de int, devemos ter um parâmetro do tipo int *, capaz de armazenar endereços de inteiros.

- A expressão “passar um vetor para uma função” deve ser interpretada como “passar o endereço inicial do vetor”. Os elementos do vetor não são copiados para a função, o argumento copiado é apenas o endereço do primeiro elemento.

Ponteiros e Alocação Dinâmica

```
float media (int n, float* v){
 int i;
 float s = 0.0;
 for (i = 0; i < n; i++)
 s += v[i];
 return s/n;
}
int main (){
 float v[10];
 float med;
 int i;
 for ( i = 0; i < 10; i++ )
 scanf("%f", &v[i]);
 med = media(10,v);
printf ( "Media = %f \n", med, var);
return 0;
```

Ponteiros e Alocação Dinâmica

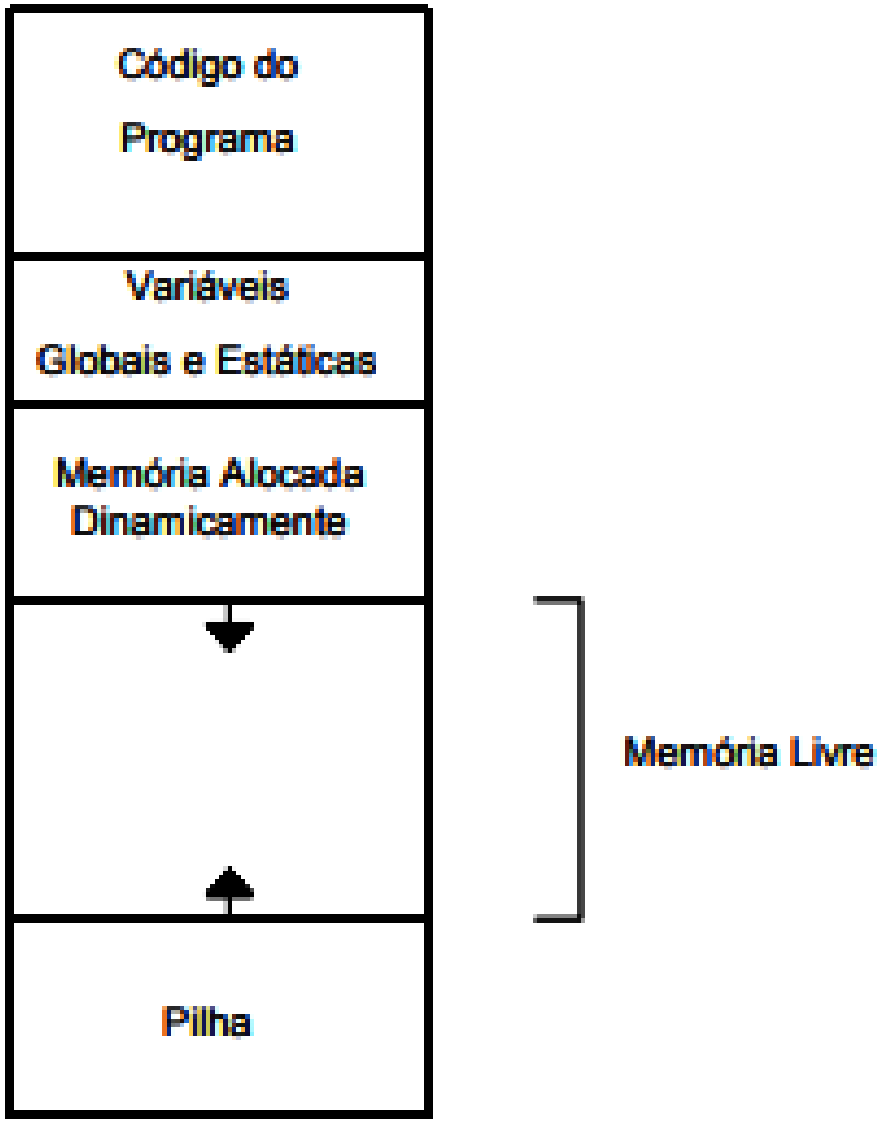
- Na declaração de um vetor, é preciso dimensioná-lo. Isto nos obrigava a saber, de antemão, quanto espaço seria necessário, isto é, tínhamos que prever o número máximo de elementos no vetor durante a codificação.

- Este pré-dimensionamento do vetor é um fator limitante.

- Se desenvolvermos um programa para calcular a média e a variância das notas de uma prova, teremos que prever o número máximo de alunos.

Ponteiros e Alocação Dinâmica

- Quando requisitamos ao sistema operacional para executar um determinado programa, o código em linguagem de máquina do programa deve ser carregado na memória



Ponteiros e Alocação Dinâmica

- A função básica para alocar memória é malloc. Ela recebe como parâmetro o número de bytes que se deseja alocar e retorna o endereço inicial da área de memória alocada.

```
int *v; v = malloc(10*4);
```

- Após este comando, se a alocação for bem sucedida, v armazenará o endereço inicial de uma área contínua de memória suficiente para armazenar 10 valores inteiros.

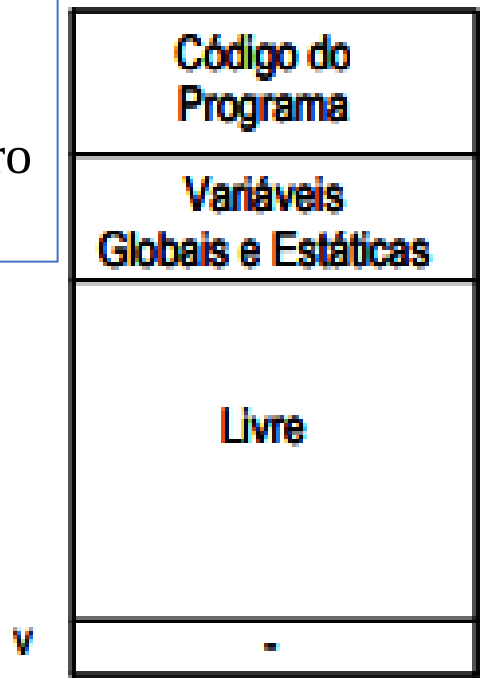
- Podemos tratar v como tratamos um vetor, pois, se v aponta para o inicio da área alocada, podemos dizer que v[0] acessa o espaço para o primeiro elemento que armazenaremos, v[1] acessa o segundo, e assim por diante até v[9].

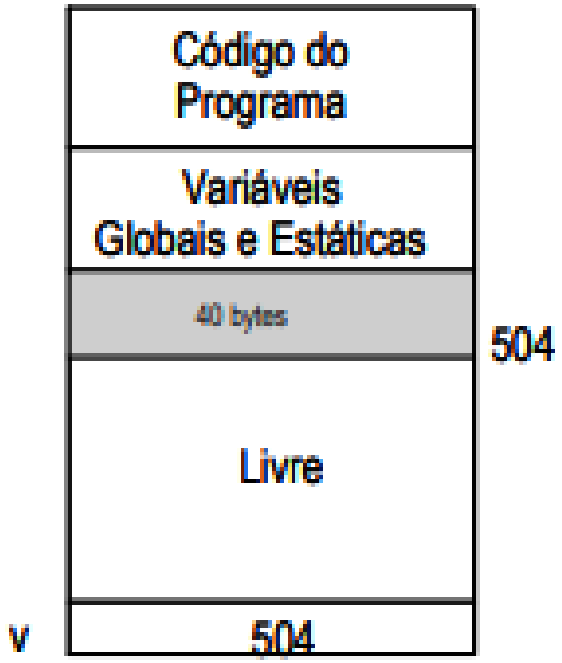
Ponteiros e Alocação Dinâmica

- Consideramos que um inteiro ocupa 4 bytes. Para ficarmos independentes de compiladores e máquinas, usamos o operador sizeof( ).

v = malloc(10*sizeof(int));

Declaração: int *v
Abre-se espaço na pilha para o ponteiro (variável local)





Comando: v = (int *) malloc (10*sizeof(int))
Reserva espaço de memória da área livre e atribui endereço à variável

Ponteiros e Alocação Dinâmica

```c
int main (){
 int i, n;
 float *v;
 float med;
 /* leitura do número de elementos */
 scanf("%d", &n);
 /* alocação dinâmica */
 v = (float*) malloc(n*sizeof(float));
 if (v==NULL) {
 printf("Memoria insuficiente.\n”);
 return 1;
 }
 /* leitura dos valores */
 for (i = 0; i < n; i++)
 scanf("%f", &v[i]);
 med = media(n,v);
 printf("Media = %f \n", med);
 /* libera memória */
 free(v);
 return 0;
```

FIM!