Ponteiros:

Na última aula vimos:
– tipos de ponteiros:

```
int * x; //ponteiro para inteiro;
char * x; //ponteiro para char;
char nome[50]; //declaração de um vetor com 50 caracteres
nome; //é um ponteiro
&nome; //endereço de memória da variável nome;
scanf("%d", &nomeVar);

void troca(int *x, int *y){
    int aux = *x;
    *x= *y;
    *y= aux;
}

void *elemento;
        └>ponteiro para tipo genérico (não especificado) de dado.

void trocaInt(void *x, void*y){
    int *a = (int *) x;//pegando o ponteiro em void e fazendo o casting pra ponteiro de inteiro
    int *b = (int *) y;
    int aux = *a;
    *a = *b;
    *b = aux;
}
//pode-se retornar um ponteiro genérico;

void trocaVet(void *vet, int i, int j, void (*troca)(void*, void*)){
    troca(&(vet[i]), &(vet[j]))
}

int main(){
    int vet[10];
     trocaVet(vet, 1, 2, trocaInt);
    return 0;
}
```

Alocação dinâmica

– Durante a execução de um programa, pode-se alocar dinamicamente memórias para usar como variáveis do programa.
– Em C, há funções específicas para essa tarefa. Tais funções estão na biblioteca *stdlib.h*.
– Funções de alocação:

- malloc
- calloc
- realloc

– Função para liberar a memória:

- free

– malloc: aloca um bloco de memória de n bytes, retornando um ponteiro para o início do bloco.

Sintaxe:

```
void * malloc (size_t n);
```

Exemplo:

```
int *vet; int tam;
printf("Digite o no. de elementos:  ");
scanf("%d", &tam);
vet = (int *) malloc(tam*sizeof(int));
```

– calloc: aloca um bloco de memória de k elementos de w bytes, zera todos os bytes alocados e retorna um ponteiro para o início do bloco.

Sintaxe:

```
void * calloc (int k, size_t w);
```

Exemplo:

```
int *vet, int tam;
printf("Digite o no. de elementos:  ");
scanf("%d", &tam);
vet = (int *)calloc(tam, sizeof(int));
```

– realloc: redimensiona um bloco de memória alocado dinamicamente.

Sintaxe:

```
void * realloc (void *ptr, unsigned int n);
```

Exemplo:

```
vet = (int *)realloc(&vet, 2*sizeof(int));
vet = 0;
```

– free: libera a memória alocada dinamicamente.

Sintaxe:

```
void free(void * ptr);
```

Exemplo:

```
free(vet);
```