Alocação Dinâmica

- Quando se declara arrays o tamanho deve ser fixo e definido primeiramente
  - Alocação estática
- Pode-se alocar dinamicamente o espaço de memória necessário a um determinado número de dados
  - Alocação mais “inteligente”

Como funciona?

- Através do uso de ponteiros
  - Se o espaço puder ser alocado o ponteiro retorna o endereço do primeiro elemento
    - A seqüência de dados pode aumentar dinamicamente
  - Em linguagem “C” a alocação pode ser feita pelas funções:
    - **malloc** e **calloc**
  - A liberação de espaço:
    - **free**
  - Se o espaço não puder ser alocado o ponteiro retorna NULL (endereço nulo)
  - Alocação feita no HEAP (restante da memória não alocada)

Utilização **MALOC**

- Alocando espaço:

```
float *pont = (float *)
  malloc(N*sizeof(float));
```

  - Onde **pont** é o nome do ponteiro

- Utilizando no programa
  - Exibindo

```
for(i=0; i<N;i++)
  printf(“elem.%d=%f\n",i,
  *(pont+i)); //pont[i]
```

  - Entrada de dados

```
for(i=0; i<N;i++)
     scanf("%d",(pont+i));
      //&pont[i]
```

Utilização
CALOC

- Alocando espaço:

```
float *pont = (float *)
  calloc(N,
  sizeof(float));
```

  - Armazena zero em todos os elementos
- Utilizando no programa
  - Exibindo

```
for(i=0; i<N;i++)
  printf(“elem.%d=%f\n",i,
  *(pont+i)); //pont[i]
```

  - Entrada de dados

```
for(i=0; i<N;i++)
     scanf("%d",(pont+i));
      //&pont[i]
```

Utilização
FREE

- Liberando espaço:

```
free(pont);
```

- Pode ser usado em qualquer lugar no programa
- Libera o espaço previamente alocado

```
#include <stdlib.h>
#include <conio.h>
void main()
{
int *ptr;
int i, N;

printf(“Quantos elementos?: ");
scanf("%d",&N);

ptr=(int *) malloc(N*sizeof(int));

for(i=0; i<N; i++)
   {
   printf("Digite elem.[%d]: ",i);
   scanf("%d", (ptr + i));
   //scanf("%d", &ptr[i]);
   }
clrscr();
for(i=N-1; i>=0; i--)
  printf("Elem.[%d]:%d\n",i,ptr[i]);

free(ptr);
}
```

```
#include <stdlib.h>
#include <conio.h>
void main()
{
int *ptr;
int i, N;
printf(“Quantos elementos?: ");
scanf("%d",&N);
ptr=(int *) malloc(N*sizeof(int));
if (ptr == NULL)
   {
   printf("Aloc.nao efetuada!!\n");
   exit(0);
   }
for(i=0; i<N; i++)
   {
   printf("Digite o elem.[%d]", i);
   scanf("%d", (ptr + i));
   }
clrscr();
for(i=N-1; i>=0; i--)
  printf("Elem.[%d]:%d\n",i,ptr[i]);
free(ptr);
}
```

```
#include <stdlib.h>
#include <conio.h>
void main()
{
int *ptr;
int i, N;
printf(“Quantos elementos?: ");
scanf("%d",&N);
ptr=(int *) malloc(N*sizeof(int));
if (ptr == NULL)
   {
   printf("Aloc.nao efetuada!!\n");
   exit(0);
   }
for(i=0; i<N; i++)
   {
   printf("Digite o elem.[%d]", i);
   scanf("%d", (ptr + i));
   }
clrscr();
for(i=N-1; i>=0; i--)
  printf("Elem.[%d]:%d\n",i,ptr[i]);
free(ptr);
}
```

```
#include <stdlib.h>
#include <conio.h>
void main()
{
int *ptr, *p;
int i, N;
printf(“Quantos elementos?: ");
scanf("%d",&N);
ptr = (int *) calloc(N,sizeof(int));
if (!ptr)
   {printf("Aloc.nao efetuada!!\n");
    exit(0);}
p =  (int *) malloc(N*sizeof(int));
if (!p)
   {printf("Aloc.nao efetuada!!\n");
    exit(0);}
printf("Valores iniciais\n");
for(i=0; i<N; i++)
   printf("p[%d]=%d,ptr[%d] = %d\n",
           i, p[i], i, ptr[i]);
free(ptr);
 free(p);
}
```