

Alocação Dinâmica de Memória

Existem duas maneiras de um programa em C armazenar dados na memória principal do computador.

A primeira, utilizando variáveis locais e globais. O que exige que o programador saiba, de antemão, a quantidade de armazenamento necessária para todas as situações a que o programa será exposto.

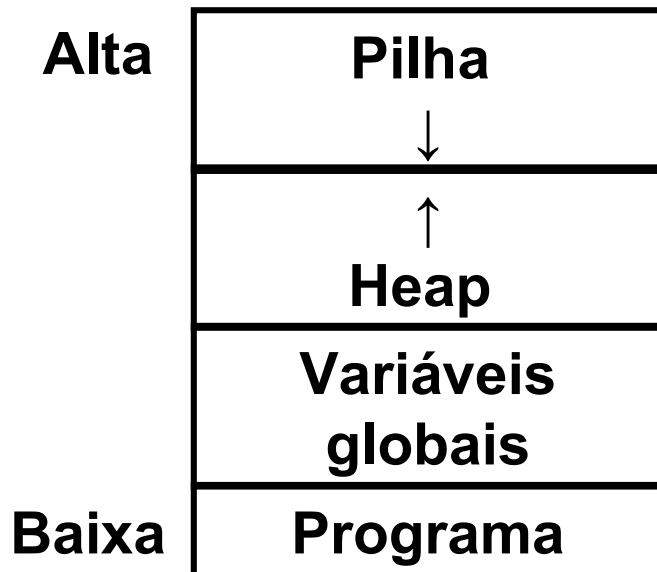
Alocação Dinâmica de Memória

Na segunda, denominada alocação dinâmica de memória, a área para o armazenamento dos dados é alocada na memória livre, também chamada de *heap*.

O *heap* fica situado entre o programa, com sua área de armazenamento permanente, e a pilha. Conforme a imagem a seguir.

Alocação Dinâmica de Memória

Memória do Sistema



A “Linguagem C” padrão ANSI define apenas 4 funções para o sistema de alocação dinâmica, disponíveis na biblioteca **stdlib.h**.

Alocação Dinâmica de Memória

- malloc

A função **malloc()** serve para alocar memória dinamicamente e tem a seguinte forma:

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

```
#include <stdio.h>
#include <stdlib.h>
main ()
{
    int *p, a, i;
    /* ... */
    p = (int *) malloc (a*sizeof(int));
    if (!p) /*A função retorna NULL se não conseguir alocar a memória solicitada*/
    {
        printf ("** Erro: Memoria Insuficiente **");
        exit (1);
    }
    for (i=0; i<a ; i++)
        p[i] = i*i;
    /* ... */
}
```

Alocação Dinâmica de Memória

Exercício:

Construa um programa que leia da entrada padrão o número de linhas e de colunas de uma matriz de floats, aloque espaço dinamicamente para esta e a inicialize, com valores fornecidos pelo usuário, através da entrada padrão. Ao final o programa deve retornar a matriz na saída padrão com layout apropriado.

```
#include <stdio.h>
#include <stdlib.h>
main () {
    int i,j,cont, cont2;
    float *matriz;
    printf ("\nEntre com o numero de linhas da matriz: ");
    scanf ("%d",&i);
    printf ("\nEntre com o numero de colunas da matriz: ");
    scanf ("%d",&j);
    matriz=(float*)malloc (i*j*sizeof(float));
    if (!matriz) {
        printf ("\nERRO!\n");
        exit (1);
    }
}
```

```
for (cont=0;cont<i*j;cont++) {  
    printf("\nEntre com o elemento da matriz[%d][%d]: ",  
        (cont/j)+1,cont%j+1);  
    scanf("%f", matriz+cont);  
}  
for (cont=0;cont<i*j;cont++)  
    if (!(cont%j))  
        printf ("| %7.2f",matriz[cont]);  
    else  
        if (cont%j==j-1)  
            printf ("%7.2f |\n", matriz[cont]);  
        else  
            printf (" %7.2f ", matriz[cont]);  
}
```

```
for (cont=0;cont<i;cont++)
    for (cont2=0;cont2<j;cont2++) {
        printf("\nEntre com o elemento da matriz[%d][%d]: ",
            cont+1, cont2+1);
        scanf("%f", matriz+cont*j+cont2);
    }
for (cont=0;cont<i*j;cont++)
    if (!(cont%j))
        printf ("| %7.2f",matriz[cont]);
    else
        if (cont%j==j-1)
            printf ("%7.2f |\n", matriz[cont]);
        else
            printf (" %7.2f ", matriz[cont]);
}
```

Alocação Dinâmica de Memória

- calloc

A função **calloc()** também serve para alocar memória. Mas, possui uma sintaxe um pouco diferente:

```
void *calloc (unsigned int num, unsigned int  
size);
```

A função aloca uma quantidade de memória igual a **num * size**, isto é, aloca memória suficiente para um vetor de **num** elementos de tamanho **size**. Uma grande diferença de **calloc** para **malloc** é que o **calloc** zera todos os bits da memória alocada.



```
#include <stdio.h>
#include <stdlib.h>
main ()
{
    int *p, a, i;
    /* ... */
    p=(int *)calloc(a,sizeof(int));
    if (!p)
    {
        printf ("** Erro: Memoria Insuficiente **");
        exit (1);
    }
    for (i=0; i<a ; i++)
        p[i] = i*i;
    ...
}
```

Alocação Dinâmica de Memória

Exercício:

Com base no que vimos, construa um programa que aloque dinamicamente memória para um vetor de strings, o número de elementos do vetor e o comprimento máximo das strings pertencentes a este, serão fornecidos pelo usuário, através da entrada padrão. O vetor deve ser inicializado, através da entrada padrão, e posteriormente, impresso na saída padrão.

```
#include <stdio.h>
#include <stdlib.h>

int main() {
    char* vetor_strings;
    int num_elementos, comprimento_max, i;
    printf ("Entre com o numero de elementos do vetor de strings: ");
    scanf ("%d",&num_elementos);
    printf ("Entre com o comprimento maximo das strings do vetor: ");
    scanf ("%d",&comprimento_max);
    vetor_strings= (char*) calloc (num_elementos, (comprimento_max+1)*sizeof(char));
    if (! vetor_strings) {
        printf ("\nERRO!\n");
        exit (1);
    }
}
```

```
for (i=0;i<num_elementos;i++)
{
    printf ("Entre com a string[%d]: ",i+1);
    scanf ("%s",vetor_strings+(i*(comprimento_max+1)));
}
printf ("\nStrings contidas no vetor:\n");
for (i=0;i<num_elementos;i++)
    printf ("\nString[%d]: %s",i+1,vetor_strings+(i*(comprimento_max+1)));
}
```