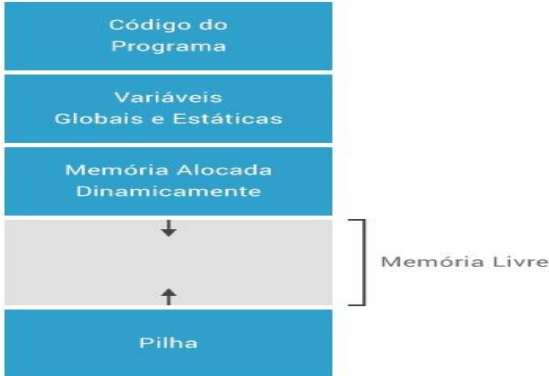




ALOCAÇÃO DINÂMICA DE MEMÓRIA		ALOCAÇÃO DINÂMICA DE MEMÓRIA
De que três maneiras o Espaço de Memória pode ser reservado no computador?		Basicamente, existem 3 maneiras de reservar espaço de memória para o armazenamento de dados: <ul style="list-style-type: none">- para o uso de variáveis Globais;- para o uso de variáveis Locais;- para o uso de Alocação Dinâmica de Variáveis;
- reservada para a questão acima -		Para as variáveis globais é reservado espaço de memória que existe enquanto o programa estiver executando. No caso das variáveis locais, o espaço existe apenas enquanto a função que declarou a variável está sendo executada, sendo liberado para outros usos quando a execução da função termina. Já a alocação dinâmica é o processo que aloca memória em tempo de execução.
O que é a Alocação Dinâmica ?		A alocação dinâmica é o processo que aloca memória em tempo de execução. Esse espaço alocado dinamicamente permanece reservado até que seja explicitamente liberado pelo programa. Ela é utilizada quando não se sabe ao certo quanto de memória será necessário para o armazenamento das informações, podendo ser determinadas em tempo de execução, conforme a necessidade do programa. Dessa forma, evita-se o desperdício de memória. A alocação dinâmica é muito utilizada em problemas de estrutura de dados como, por exemplo, listas encadeadas, pilhas, filas, árvores binárias e grafos.
Ilustre como a distribuição de memória ocorre num computador...		Na Figura abaixo é possível observar, de maneira fictícia, a distribuição do uso da memória pelo sistema operacional. Quando um programa é executado, o sistema operacional reserva espaços de memória necessários para armazenar as variáveis globais (estáticas). O restante da memória livre é utilizado pelas variáveis locais e pelas variáveis alocadas dinamicamente. Cada vez que uma determinada função é chamada, o sistema reserva o espaço necessário para as variáveis locais da função. Esse espaço pertence à pilha de execução e, quando a função termina, é desempilhado...
- reservada para a questão acima -		
- reservada para a questão acima -		
Como podemos trabalhar com Alocação de Memória Dinâmica na Linguagem C ?		Na linguagem C, para trabalhar com a alocação dinâmica de memória, é necessário utilizar funções da biblioteca "stdlib.h", essas funções são: <ul style="list-style-type: none">- malloc()- sizeof()- realloc()- free()
O que faz a função malloc ?		Função malloc: A função malloc reserva espaço de memória livre de tipo genérico, ou seja, pode ser de qualquer tipo de dado. Esta função recebe como parâmetro o número de bytes de memória que se deseja alocar e retorna um ponteiro (endereço) para se houver espaço livre, caso contrário, retorna um endereço nulo (representado pelo símbolo NULL), se não houver espaço livre. A sintaxe da função é a seguinte: void * malloc(number_bytes); //número de bytes alocados do formato inteiro sem sinal (void pode ser qualquer tipo de dado, ou o void mesmo para tipo genérico)
Exemplifique como podemos usar a função malloc num caso real...		Podemos usar como exemplo a alocação dinâmica de um vetor de inteiro com 20 elementos. Como a função malloc tem como valor de retorno o endereço da área alocada e, nesse exemplo, desejamos armazenar valores inteiros nessa área, devemos declarar um ponteiro de inteiro para receber o endereço inicial do espaço alocado. Abaixo, segue a codificação... int *v; v = malloc (20*4); //esse valor 4 é de 4 bytes - 20 vezes 4 bytes
- reservada para a questão acima -		Após esse comando, se a alocação for bem-sucedida, "v" armazenará o endereço inicial de uma área contínua de memória suficiente para armazenar 20 valores inteiros. Nesse caso, podemos tratar o vetor "v" como se ele estivesse declarado estaticamente, pois se "v" aponta para o início da área alocada, então "v[0]" acessa o espaço para o primeiro elemento armazenado, "v[1]" acessa o segundo, e assim sucessivamente.

ALOCÇÃO DINÂMICA DE MEMÓRIA		ALOCÇÃO DINÂMICA DE MEMÓRIA
Como os espaços de memória reservados por através da função malloc geralmente são genéricos, o que podemos utilizar para reservar um espaço de memória de um tipo de dado específico ?		<p>Para fazer isso, comum fazer a conversão explicitamente, utilizando o operador de molde de tipo (cast). A linha de instrução para a alocação do vetor de inteiros fica então:</p> <p><i>v = (int *) malloc (20*4);</i></p> <p>No (int *) ocorreu o cast para valores inteiros.</p>
O que faz a função sizeof ?		<p>Função sizeof: retorna o tamanho, em bytes, do que for definido como parâmetro para a função. A sintaxe da função é a seguinte:</p> <p><i>sizeof (variavel ou tipo de dados);</i></p>
Exemplifique como podemos usar a função sizeof num caso real...		<p>Na alocação do vetor com 20 números inteiros mostrado anteriormente, consideramos que um inteiro ocupa 4 bytes. Para ficarmos independentes de compiladores e máquinas, usamos o operador sizeof(), como mostrado no exemplo a seguir...</p> <p><i>v = (int *) malloc (20*sizeof(int));</i></p> <p>Isso faz com que cada valor índice ocupe o tamanho determinado para um inteiro na Linguagem C</p>
Ilustre como a alocação de memória por através das funções malloc e sizeof acontecem na memória do computador...		<div><div><p>1 – Declaração: int*v Abre-se espaço na pilha para o ponteiro (variável local)</p></div><div><p>2 – Comando: v = (int*) malloc (10*sizeof(int)) Reserva espaço de memória da área livre e atribui endereço à variável</p></div></div>
- reservada para a questão acima -		
O que faz a função realloc ?		<p>Função realloc: é usada para redimensionar um espaço alocado previamente com malloc. Seus argumentos são um ponteiro para o início de uma área previamente alocada pelo malloc, e o novo tamanho, que pode ser maior ou menor que o tamanho original. realloc vai retornar um ponteiro para a nova área alocada. Este ponteiro pode ser igual ao ponteiro original se o novo tamanho for menor que o tamanho original, e diferente do ponteiro original se o novo tamanho for maior que o tamanho original. Neste último caso, realloc copia os dados da área original para a nova área. A sintaxe da função é a seguinte:</p> <p><i>void * realloc (void *p, tamanho_novo);</i></p>
Exemplifique como podemos usar a função realloc num caso real...		<p>No exemplo abaixo, é mostrada a alocação dinâmica, utilizando a função malloc, da variável frase como uma string de tamanho 6. Em seguida, com a função realloc, é feita a realocação da variável frase com tamanho 20...</p> <p><i>char *frase;</i> <i>frase=(char *)malloc(6);</i></p> <p><i>strcpy(frase,"teste");</i></p> <p><i>//realocação da variável frase na memoria</i> <i>frase=(char *)realloc(frase,20);</i></p> <p><i>strcpy(frase,"teste estrutura");</i></p>
- reservada para a questão acima -		
O que faz a função free ?		<p>Função free: As variáveis alocadas dentro de uma função, também conhecidas como variáveis automáticas ou locais, desaparecem assim que a execução da função termina. Já as variáveis alocadas dinamicamente continuam a existir mesmo depois que a execução da função termina. Nestes casos é necessário liberar a memória ocupada por essas variáveis, para isso usamos a função free. Ela recebe como argumento um ponteiro para uma área de memória previamente alocada por malloc() e então libera esta área para uma possível utilização futura. Aplique free apenas ao bloco todo. Sua sintaxe é:</p> <p><i>free (variavel);</i></p>
Exemplifique como podemos usar a função free num caso real...		<p>No exemplo abaixo, é mostrada a alocação dinâmica, utilizando a função malloc, da variável teste e, em seguida, o espaço reservado para a variável é liberado, utilizando a função free.</p> <p><i>float *teste;</i></p> <p><i>teste = (float *) malloc (15*sizeof(float));</i></p> <p><i>free (teste);</i></p>