PROGRAMAÇÃO DE COMPUTADORES

PROFESSOR: FÁBIO GARCEZ BETTIO

ESTUDANTE: CLÍSTENES GRIZAFIS BENTO

APS 2 ATIVIDADE ALOCAÇÃO DINÂMICA (SLIDE 37 A 42)

1. Aloque uma área de memória com malloc para armazenar um número inteiro. Insira nesta área de memória o número 27 e imprima.
Incremente o número 27 usando o ponteiro e imprima o resultado.
Desaloque o endereço alocado.

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

/*1. Aloque uma área de memória com malloc para armazenar um número inteiro. Insira nesta área de memória o número 27 e imprima.
Incremente o número 27 usando o ponteiro e imprima o resultado.
Desaloque o endereço alocado. */


int main ()
{
	setlocale(LC_ALL,"portuguese");

	int *numero;

	numero=(int*)malloc(sizeof(int));

	*numero=27;

	printf("O número inteiro alocado dentro do ponteiro é: %d", *numero);
	printf("\nA posisão do ponteiro é: %X", &numero);

	free(numero);

	printf("\n\n\n");

	system("pause");

	return 0;

}
```

2. Aloque 5 áreas de memória com malloc para armazenar 5 números inteiro, cada alocação deve ser colocada em um vetor de ponteiros
int *vet[5];
Imprima os valores usando o vetor de ponteiros.

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

/* 2. Aloque 5 áreas de memória com malloc para armazenar 5 números inteiro, cada alocação deve ser colocada em um vetor de ponteiros int *vet[5];
Imprima os valores usando o vetor de ponteiros. */

int main ()
{
	setlocale(LC_ALL,"portuguese");

	int *numero[5];
	int i=0,j=0, k=0;

	printf("SEJA BEM VINDO");

	for(i=0;i<5;i++)
	{

		numero[i]=(int*)malloc(sizeof(int));

	}

	for(j=0;j<5;j++)

	{
		printf("\nDigite o valor para o ponteiro de posição %d: ",j+1);

		scanf("%d",numero[j]);
	}

	for(k=0;k<5;k++)
	{

		printf("\nO número inteiro alocado dentro do ponteiro %d é: %d",k+1,*numero[k]);
		printf("\nA posisão do ponteiro %d é: %X",k+1, &numero[k]);

	}

	free(numero);
	printf("\n\n\n");

	system("pause");

	return 0;

}
```

3. Aloque uma área de memória com malloc para armazenar uma struct com nome e telefone (ambos vetor de char).
Preencha os dados.
Imprima os dados.

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>
#include <string.h>

/*3. Aloque uma área de memória com malloc para armazenar uma struct com nome e telefone (ambos vetor de char).
Preencha os dados.
Imprima os dados.
 */

struct CONTATO
{
	char nome[30];
	char telefone[30];
};


int main ()
{
	setlocale(LC_ALL,"portuguese");

	struct CONTATO *pessoa;

	pessoa=(struct CONTATO*)malloc(sizeof(struct CONTATO));

	printf("\n\nPor gentileza digite o nome do contato: ");
	scanf("%s",pessoa->nome);
	printf("\n\nAgora digite o telefone do contato: ");
	scanf("%s",pessoa->telefone);

	printf("O nome da pessoa alocado dentro do ponteiro é: %s", pessoa->nome);
	printf("\n\nA posisão do ponteiro é: %X", &pessoa->nome);
	printf("\nO nome da pessoa alocado dentro do ponteiro é: %s", pessoa->telefone);
	printf("\nA posisão do ponteiro é: %X", &pessoa->telefone);

	free(pessoa);

	printf("\n\n\n");

	system("pause");

	return 0;

}
```

4. Crie um programa em C que preencha uma struct CADASTRO com duas subtructs ENDERECO (comercial e residencial) usando scanf, depois do preenchimento imprima todo seu conteúdo. Sem alocação estática, apenas ponteiros e malloc.

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

/* 4.Crie um programa em C que preencha uma struct CADASTRO com duas subtructs ENDERECO (comercial e residencial) usando scanf, depois do preenchimento imprima todo seu conteúdo. Sem alocação estática, apenas ponteiros e malloc.
 */



struct ENDERECO //substruct
{
	char rua [50]; //variável que vai receber o nome da rua
	char numero[50]; //variável que vai receber o número da rua

};

struct CADASTRO //a struct principal
{
	char nome [30];
	struct ENDERECO res; // res = residencial
	struct ENDERECO com; // com = comercial
};



int main()
{
	setlocale(LC_ALL,"portuguese"); //coloca idioma

	struct CADASTRO *pessoa; //declaração do ponteiro struct cadastro com a variável "pessoa"

	pessoa=(struct CADASTRO*)malloc(sizeof(struct CADASTRO)); //alocando espaço na memória para a struct

	printf("BEM VINDO AO CADASTRO DE ENDEREÇOS\n"); // mensagem de boas vindas
	printf("NESSE PROGRAMA VOCÊ PODE CADASTRAR SEU ENDEREÇO RESIDENCIAL E COMERCIAL\n\n");

	printf("Por gentileza, digite o seu nome: ");
	scanf("%s",pessoa->nome); //leitura de do nome
	printf("\n Por gentileza digite o seu endereço residencial: ");
	scanf("%s",pessoa->res.rua); //leitura da rua residencial
	printf("\n Agora o número: ");
	scanf("%s",pessoa->res.numero); //leitura do número residencial
	printf("\n Por gentileza digite o seu endereço comercial: ");
	scanf("%s",pessoa->com.rua); //leitura do rua comercial
```

```
        printf("\n Agora o número: ");
        scanf("%s",pessoa->com.numero); //leitura do número comercial



        system("cls"); // limpa a tela

        /*impressão das informações */

        printf("NOME: %s, posição do ponteiro: %X",pessoa->nome, &pessoa->nome);
        printf("\nENDEREÇO RESIDENCIAL: %s, %s",pessoa->res.rua,pessoa->res.numero);
        printf("\nENDEREÇO COMERCIAL: %s, %s",pessoa->com.rua,pessoa->com.numero);

        free(pessoa); //liberando espaço na memória

        printf("\n\n\n");

        system("pause"); //pausa o programa
        return 0;

}
```

5. Crie duas structs, uma estática e outra dinâmica com os seguintes campos
Quantidade;
Descrição do produto;
Valor Unitário;
Preencha a struct estática com scanf;
Aloque a struct dinâmica
Copie os dados a struct estática para a dinâmica;
Imprima a struct dinâmica.

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>
#include <string.h>

/* 5.Crie duas structs, uma estática e outra dinâmica com os seguintes campos
Quantidade;
Descrição do produto;
Valor Unitário;
Preencha a struct estática com scanf;
Aloque a struct dinâmica
Copie os dados a struct estática para a dinâmica;
Imprima a struct dinâmica.
*/

struct PRODUTO1
{

	int quantidade;
	char descricao_produto[50];
	float valor_unitario=0;

};

struct PRODUTO2
{

	int quantidade;
	char descricao_produto[50];
	float valor_unitario=0;

};

int main ()
{

	setlocale(LC_ALL,"portuguese");

	struct PRODUTO1 produto_estatico;
	struct PRODUTO2 *produto_dinamico;

	produto_dinamico = (struct PRODUTO2*)malloc(sizeof(struct PRODUTO2));
```

```
        printf("BEM VINDO");

        printf("\n\nPor gentileza digite a quantidade do produto: ");
        scanf("%d",&produto_estatico.quantidade);
        printf("\Agora descreva o produto: ");
        scanf("%s",&produto_estatico.descricao_produto);
        printf("\nDigite o valor do produto em R$: ");
        scanf("%f",&produto_estatico.valor_unitario);

        produto_dinamico->quantidade=produto_estatico.quantidade;
        strcpy(produto_dinamico->descricao_produto,produto_estatico.descricao_produto);
        produto_dinamico->valor_unitario=produto_estatico.valor_unitario;

        printf("\n\nA quantidade do produto é: %d", produto_dinamico->quantidade);
        printf("\nA descrição do produto é: %s", produto_dinamico->descricao_produto);
        printf("\nO valor unitário do produto é: R$%.2f", produto_dinamico->valor_unitario);

        free(produto_dinamico);

        printf("\n\n\n");

        system("pause");

        return 0;

}
```

6. Crie um programa em C que preencha 5 structs CADASTRO com nome e telefone usando scanf,
 cada endereço retornado pelo malloc será armazenado em um vetor de ponteiros.

struct CADASTRO *vet[5];

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>
#include <string.h>

/* 6. Crie um programa em C que preencha 5 structs CADASTRO com nome e telefone usando scanf,
 cada endereço retornado pelo malloc será armazenado em um vetor de ponteiros.

struct CADASTRO *vet[5];
*/

struct CADASTRO
{

	char nome[50];
	int telefone=0;

};


int main ()
{

	setlocale(LC_ALL,"portuguese");

	struct CADASTRO *pessoa[5];
	int i=0,j=0, k=0;

	printf("SEJA BEM VINDO");

	for(i=0;i<5;i++)
	{

		pessoa[i]=(struct CADASTRO*)malloc(sizeof(struct CADASTRO));

	}

	for(j=0;j<5;j++)

	{
		printf("\n\nDigite o nome da pessoa [%d]: ",j+1);
		scanf("%s",pessoa[j]->nome);

		printf("\nDigite o telefone da pessoa [%d]: ",j+1);
		scanf("%d",&pessoa[j]->telefone);
	}
```

```
	system("cls");

	for(k=0;k<5;k++)
	{

		printf("\n\n\nO nome da pessoa [%d] é: %s",k+1,pessoa[k]->nome);
		printf("\nO telefone da pessoa [%d] é: %d",k+1, pessoa[k]->telefone);

	}

	free(pessoa);

	printf("\n\n\n");

	system("pause");

	return 0;

}
```

Arquivo com programas criados: https://drive.google.com/open?id=1vkRqOormsFeO20i3oTreMsZZybNDGLhh