Alocação dinâmica de memória

Programação de Computadores I
Universidade Federal de Ouro Preto

Laboratório

Exercício 1

Crie um vetor dinâmico de tamanho informado pelo usuário.
Solicite os valores do vetor ao usuário.
Em seguida, imprima o conteúdo do vetor.

Resposta

```
int main()
{
    int *ptr;
    int tam, i;

    printf("Informe quantos elementos deseja armazenar:");
    scanf("%d", &tam);

    ptr = malloc(tam * sizeof(int));
    if(!ptr) {
        printf("Nao ha memoria.\n");
        exit(0);
    }

    for(i = 0; i < tam; ++i) {
        printf("Informe o valor da posicao %d\n", i);
        scanf("%d", &ptr[i]);
    }

    if(tam > 0)
        for(i = 0; i < tam; ++i) {
        printf("v[%d] = %d\n", i,ptr[i]);
    }

    free(ptr);
    return 0;
}
```

Exercício 2

Converta o código do exercício anterior de forma a termos:

- procedimento para preencher o vetor, solicitando os valores ao usuário.
- procedimento para imprimir o vetor.
- função para efetuar a alocação dinâmica do vetor, retornando o endereço alocado para o mesmo.

Resposta

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

int* aloca(int);
void preencheVetor(int*,int);
void imprimeVetor(int*,int);

int main()
{
    int *ptr;
    int tam, i;

    printf("Informe quantos elementos deseja
armazenar:");
    scanf("%d", &tam);

    ptr = aloca(tam);
    preencheVetor(ptr,tam);
    imprimeVetor(ptr,tam);

    free(ptr);

    return 0;
}

int* aloca(int tam){
    int *ptr = malloc(tam * sizeof(int));
    if (ptr!=NULL)
        return ptr;
    else{
        printf("Nao ha memoria.\n");
        exit(0);
    }
}

void preencheVetor(int* vet, int tam){
    int i;
    for(i = 0; i < tam; ++i) {
        printf("Informe o valor da posicao
%d\n", i);
        scanf("%d", &vet[i]);
    }

}
void imprimeVetor(int* vet,int tam){
    int i;
    if(tam > 0)
        for(i = 0; i < tam; ++i) {
        printf("v[%d] = %d\n", i,vet[i]);
    }
}
```

Exercício 3

Crie uma matriz dinâmica de tamanho l linhas e c colunas.
Solicite o preenchimento do seu conteúdo pelo usuário.
Em seguida, imprima a matriz.
Dica: utilize l=2 e c=3 para testar seu programa!

Resposta

```
int main() {

    float **matriz;
    int l, c, i, j;
    printf("Digite numero de linhas da
matriz:");
    scanf("%d",&l);
    printf("Digite numero de colunas da
matriz:");
    scanf("%d",&c);

    /* aloca as linhas da matriz */
    matriz = (float **) malloc(l *
sizeof(float *));
    if(matriz == NULL) {
        printf("** Erro: memoria
insuficiente **");
        return NULL;
    }
    /* aloca as colunas da matriz */
    for(i = 0; i < l; i++) {
        matriz[i] = (float*) malloc (c *
sizeof(float));
        if (matriz[i] == NULL) {
            printf ("** Erro: Memoria
Insuficiente **");
            return NULL;
        }
    }

    for(i = 0; i < l; ++i) {
        for(j = 0; j < c; ++j) {
            printf( "Informe
elemento[%d][%d]:\n", i, j );
            scanf("%f", &matriz[i][j]);
        }
    }

    for(i = 0; i < l; ++i) {
        for(j = 0; j < c; ++j)
            printf( "%10.2f", matriz[i][j]
);
        printf("\n");
    }

    return 0;
}
```

Exercício 4 - Para casa

Converta o código do exercício anterior de forma a termos:

- procedimento para preencher a matriz, solicitando os valores ao usuário.
- procedimento para imprimir a matriz.
- função para efetuar a alocação dinâmica da matriz, retornando o endereço alocado para a mesma.

Resposta

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

int preencheMatriz(float **m, int l, int c);
void imprimeMatriz(float **m, int l, int c);
float ** alocarMatriz(int, int);

int main() {

    float **matriz;
    int l, c, i, j;
    printf("Digite numero de linhas da matriz:");
    scanf("%d",&l);
    printf("Digite numero de colunas da matriz:");
    scanf("%d",&c);

    matriz = alocarMatriz(l, c);
    preencheMatriz(matriz,l,c);
    imprimeMatriz(matriz,l,c);

    free(matriz);

    return 0;
}


int preencheMatriz(float **m, int l, int c){
    int i, j;
    for(i = 0; i < l; ++i) {
        for(j = 0; j < c; ++j) {
            printf( "Informe elemento[%d][%d]:\n", i, j );
            scanf("%f", &m[i][j]);
        }
    }
}

void imprimeMatriz(float **m, int l, int c){
    int i, j;
    for(i = 0; i < l; ++i) {
        for(j = 0; j < c; ++j)
            printf( "%10.2f", matriz[i][j] );
        printf("\n");
    }
}

float **alocarMatriz(int l, int c){
    float **m;
    int i;

    /* aloca as linhas da matriz */
    m = (float **) malloc(l * sizeof(float *));
    if(m == NULL) {
        printf("** Erro: memoria insuficiente **");
        return NULL;
    }
    /* aloca as colunas da matriz */
    for(i = 0; i < l; i++) {
        m[i] = (float*) malloc (c * sizeof(float));
        if (m[i] == NULL) {
            printf ("** Erro: Memoria Insuficiente **");
            return NULL;
        }
    }
    return m; /* retorna o ponteiro para a matriz */
}
```