Arranjo multidimensional - exercício

Roberto Rocha

PUC Minas Virtual

Alocação dinâmica de memória – Matrizes – Exercício

Desenvolver um programa que leia o número de alunos em uma turma.
Em seguida:
a) criar uma função que devolva um vetor com os nomes dos alunos.
b) uma função que devolva uma matriz contendo quatro notas por aluno – notas do tipo inteiro.
c) uma função que receba a matriz de notas e devolva um vetor do tipo real contendo a média de cada aluno.
d) uma função que devolva um vetor com a classificação em ordem alfabética dos alunos da turma.
e) uma função que receba os vetores e matrizes criados e imprima os alunos em ordem alfabética , bem como suas médias

0	J	O	S	E		
1	M	A	R	I	A	
2	J	O	A	O		
3	A	N	A			
4	L	I	V	I	A	

N1	N2	N3	N4
5	6	6	8
7	6	8	5
8	8	8	8
2	8	9	7

media
6.25
6.5
8.0
6.5

índice
3
2
0
4
1

Exercício

Desenvolver um programa que leia o número de alunos em uma turma.
Em seguida:
a) criar uma função que devolva um vetor com os nomes dos alunos.
b) uma função que devolva uma matriz contendo quatro notas por aluno – notas do tipo inteiro.
c) uma função que receba a matriz de notas e devolva um vetor do tipo real contendo a média de cada aluno.
d) uma função que devolva um vetor com a classificação em ordem alfabética dos alunos da turma.
e) uma função que receba os vetores e matrizes criados e imprima os alunos em ordem alfabética , bem como suas médias

Como não sabemos o número de alunos, vamos trabalhar com alocação dinâmica de memória!

a) criar uma função que devolva um vetor com os nomes dos alunos.

```
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>
#include <string.h>
int main()
{
   setlocale(LC_ALL,"portuguese");
  int nrAlunos;
  char **nomeAlunos;
  printf("Digite o Número de alunos:");
  scanf("%d",&nrAlunos);
  nomeAlunos=lerAlunos(nrAlunos);
```

Teremos um vetor de vetores!

J	O	S	E		
M	A	R	I	A	
J	O	A	O		
A	N	A			
L	I	V	I	A	

```
char ** lerAlunos(int nrAlunos)
{
   char **alunos;
   int i;
   alunos=malloc(sizeof(char)*nrAlunos);
   // para cada linha será criada um espaço para armazenar os nomes
   // vamos considerar 50 posições para cada nome de aluno
   for (i=0;i<nrAlunos;i=i+1)
   {
      alunos[i]=malloc(sizeof(char)*50);
   }
   for (i=0;i<nrAlunos;i=i+1)
   {
      printf("Digite o nome do %d aluno:",i+1);
      fflush(stdin);
      gets(alunos[i]);
   }
   return alunos;
}
```

b) uma função que devolva uma matriz contendo quatro notas por aluno – notas do tipo inteiro.

```
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>
#include <string.h>
const int QTDNOTAS=4;
int main()
{
   setlocale(LC_ALL,"portuguese");
  int nrAlunos;
  char **nomeAlunos;
  int **notas;
  printf("Digite o Número de alunos:");
  scanf("%d",&nrAlunos);
  nomeAlunos=lerAlunos(nrAlunos);
  notas=leNotas(nrAlunos,nomeAlunos);
```

Constante QTDNOTAS para caso o número de notas variar alternar apenas o valor.

Teremos um vetor de vetores!

N1	N2	N3	N4
5	6	6	8
7	6	8	4
8	8	8	8
2	8	9	7

```
int ** leNotas(int nrAlunos, char ** alunos)
{
   int **notasAlunos;
   int i,j;
   notasAlunos=malloc(sizeof(int)*nrAlunos);
   //em linha será criada QTDNOTAS colunas para armazenar as notas
   for(i=0;i<nrAlunos;i=i+1)
   {
      notasAlunos[i]=malloc(sizeof(int)*QTDNOTAS);
   }
   //lendo as notas
   for (i=0;i<nrAlunos;i=i+1)
   {
      printf("Aluno: %s\n ",alunos[i]);
      for (j=0;j<QTDNOTAS;j=j+1)
      {
         printf("digite a %d nota:",j+1);
         scanf("%d",&notasAlunos[i][j]);
      }
   }
   return notasAlunos;
}
```

c) uma função que receba a matriz de notas e devolva um vetor do tipo real contendo a média de cada aluno.

```
int main()
  {
     setlocale(LC_ALL,"portuguese");
    int nrAlunos;
    char **nomeAlunos;
    int **notas;
    float *media;
    printf("Digite o Número de alunos:");
    scanf("%d",&nrAlunos);
    nomeAlunos=lerAlunos(nrAlunos);
    notas=leNotas(nrAlunos,nomeAlunos);
    media=calculaMedia(nrAlunos,notas);
```

Constante QTDNOTAS para caso o número de notas variar alternar apenas o valor.

Teremos um vetor com as médias

N1
5.2
7.3
8.0
4.5

```
float * calculaMedia(int nrAlunos,int **notas)
{
   float *mediaNotas;
   int i,j;
   float soma,media;
   mediaNotas=malloc(sizeof(float)*nrAlunos);
   for (i=0;i<nrAlunos;i=i+1)
   {
      soma=0;
      for (j=0;j<QTDNOTAS;j=j+1)
      {
         soma=soma+(float) notas[i][j];
      }
      media=soma/QTDNOTAS;
      mediaNotas[i]=media;
   }
   return mediaNotas;
}
```

d) uma função que devolva um vetor com a classificação em ordem alfabética dos alunos da turma.

```
int main()
  {
     setlocale(LC_ALL,"portuguese");
    int nrAlunos;
    char **nomeAlunos;
    int **notas;
    float *media;
    int *classificacao;
    printf("Digite o Número de alunos:");
    scanf("%d",&nrAlunos);
    nomeAlunos=lerAlunos(nrAlunos);
    notas=leNotas(nrAlunos,nomeAlunos);
    media=calculaMedia(nrAlunos,notas);
classificacao = classificaAlunos(nrAlunos,nomeAlunos);
```

0	J	O	S	E		
1	M	A	R	I	A	
2	J	O	A	O		
3	A	N	A			
4	L	I	V	I	A	

Teremos um vetor com os índices em ordem alfabética

índice
3
2
0
4
1

d) uma função que devolva um vetor com a classificação em ordem alfabética dos alunos da turma.

Teremos um vetor com os índices em ordem alfabética

0	J	O	S	E		
1	M	A	R	I	A	
2	J	O	A	O		
3	A	N	A			
4	L	I	V	I	A	

índice
3
2
0
4
1

int strcmp(const char *str1, const char *str2)

se Return value < 0 então str1 é menor que str2.
se Return value > 0 então str1 é maior que str2.
se Return value = 0 então str1 é igual a str2.

```
int * classificaAlunos(int nrAlunos,char **nomeAlunos)
{
   int i,j,aux;
   int * nomeClassificado;// irá armazenar apenas o indice do nome.
   // utilizando o método de classificação
   nomeClassificado=malloc(sizeof(int)*nrAlunos);
   for (i=0;i<nrAlunos;i=i+1)
   {
      nomeClassificado[i]=i;
   }

   for (i=0;i<nrAlunos;i=i+1)
   {
      for (j=i+1;j<nrAlunos;j=j+1)
      {
         if (strcmp(nomeAlunos[nomeClassificado[i]],nomeAlunos[nomeClassificado[j]])>0)
         {
            aux=nomeClassificado[i];
            nomeClassificado[i]=nomeClassificado[j];
            nomeClassificado[j]=aux;
         }
      }
   }
   return nomeClassificado;
}
```

e) uma função que receba os vetores e matrizes criados e imprima os alunos em ordem alfabética , bem como suas médias

```
int main()
  {
     setlocale(LC_ALL,"portuguese");
    int nrAlunos;
    char **nomeAlunos;
    int **notas;
    float *media;
    int *classificacao;
    printf("Digite o Número de alunos:");
    scanf("%d",&nrAlunos);
    nomeAlunos=lerAlunos(nrAlunos);
    notas=leNotas(nrAlunos,nomeAlunos);
    media=calculaMedia(nrAlunos,notas);
    classificacao = classificaAlunos(nrAlunos,nomeAlunos);
    imprimeBoletimFinal (nrAlunos, nomeAlunos, classificacao,notas,media);

  return 0;
}
```

e) uma função que receba os vetores e matrizes criados e imprima os alunos em ordem alfabética , bem como suas médias

```
void imprimeBoletimFinal (int nrAlunos, char **alunos, int *classificado, int **notas, float *media)
{
   int i,j;
   printf("Notas Finais em ordem alfabética de nome de aluno\n");
   printf("Ordem \t nome");
   for (i=0;i<QTDNOTAS;i=i+1)
   {
      printf("\t n%d",i+1);
   }
   printf("\t media \n");
   for (i=0;i<nrAlunos;i=i+1)
   {
      printf("%d\t%s",i+1,alunos[classificado[i]]);
      for (j=0;j<QTDNOTAS;j=j+1)
      {
         printf("\t%2d",notas[classificado[i]][j]);
      }
      printf("\t%6.2f\n",media[classificado[i]]);
   }
}
```

PUC Minas
Virtual