Introdução a estruturas heterogêneas (structs) e arranjos de estruturas.

Programação de Computadores I
Universidade Federal de Ouro Preto

Exercício 1

Crie uma struct para representar as partes reais e imaginárias de um número complexo. Seu programa deve solicitar dois números complexos e apresentar o resultado da soma de ambos.

Exercício 1 - Resposta - `main.c`

```c
#include <stdio.h>
#include "complexo.h"

int main()
{
    NumComplexo num1, num2, numSoma;

    printf("Informe a parte real do primeiro numero: ");
    scanf("%f", &num1.real);
    printf("Informe a parte imaginaria do primeiro numero: ");
    scanf("%f", &num1.imaginaria);
    printf("Informe a parte real do segundo numero: ");
    scanf("%f", &num2.real);
    printf("Informe a parte imaginaria do segundo numero: ");
    scanf("%f", &num2.imaginaria);

    numSoma = somar(num1, num2);
    printf("A soma e:\n");
    imprimir(numSoma);

    return 0;
}
```

Exercício 1 - Resposta - `complexo.c`

```
#include <stdio.h>
#include "complexo.h"

NumComplexo somar(NumComplexo n1, NumComplexo n2)
{
    NumComplexo resultado;
    resultado.real = n1.real + n2.real;
    resultado.imaginaria = n1.imaginaria + n2.imaginaria;
    return resultado;
}

void imprimir(NumComplexo n)
{
     printf("(%.2f) + (%.2f)i\n", n.real, n.imaginaria);
}
```

Exercício 1 - Resposta - `complexo.h`

```
#ifndef COMPLEXO_H
#define COMPLEXO_H

struct NumComplexo
{
    float real;
    float imaginaria;
};
typedef struct NumComplexo NumComplexo;

NumComplexo somar(NumComplexo, NumComplexo);
void imprimir(NumComplexo);

#endif
```

Exercício 2

Crie uma estrutura para representar um grupo de alunos. Cada registro deve possuir o campo nome e três notas. Solicite que o usuário preencha as informações e, em seguida, imprima todos os dados dos respectivos alunos.

Exercício 2 - Resposta

```
typedef struct aluno{
   char nome [40];
   double nota[4];
} Aluno;

void leVetorAluno(Aluno temp[], int tam){
   int i, j;
   for (i = 0; i < tam; ++i){
       printf("Informe o nome do aluno: ");
       fgets(temp[i].nome, 40, stdin);
       for (j = 0; j < 4; ++j){
           char leitura[15];
           printf("Informe a nota %d: ",j+1);
           fgets(leitura, 15, stdin);
           double notaLida = atof(leitura);
           temp[i].nota[j] = notaLida;
       }
   }
}
```

```
void imprimeVetorAluno(Aluno temp[], int tam){
   int i, j;
   for (i = 0; i < tam; ++i){
       printf("Aluno %d:", i+1);
       printf("\tNome: %s\n",temp[i].nome);
       for (j = 0; j < 4; ++j){
           printf("\tNota %d = %.2lf\n",
                  j+1, temp[i].nota[j]);
       }
   }
}
int main(){

   Aluno alunos[2];
   leVetorAluno(alunos, 2);
   imprimeVetorAluno(alunos, 2);
   return 0;
}
```

Exercício 3

Crie uma estrutura para representar um grupo de alunos. Desta vez, solicite que o usuário informe quantos alunos devem ser criados e aloque-os dinamicamente. Cada registro deve possuir o campo nome e três notas. Solicite que o usuário preencha as informações e, em seguida, imprima todos os dados dos respectivos alunos.

Exercício 3 - Resposta

```
typedef struct aluno{
    char nome [40];
    double nota[4];
} Aluno;

void leVetorAluno(Aluno*  temp, int tam){

    int i,j;
    for (i = 0; i < tam; ++i){
        fgets(temp[i].nome,  40, stdin);
        for (j = 0; j < 4; ++j){
            char leitura[15];
            fgets(leitura, 15, stdin);
            double notaLida = atof(leitura);
            temp[i].nota[j]  = notaLida;
        }
    }
}
```

```
void imprimeVetorAluno(Aluno*  temp, int tam){

    int i,j;
    for (i = 0; i < tam; ++i){
        printf("Aluno  %d:\n", i+1);
        printf("\tNome:  %s", temp[i].nome);
        for (j = 0; j < 4; ++j){
            printf("\tNota  %d = %.2lf\n", j+1,
temp[i].nota[j]);
        }
    }
}
int main(){

    Aluno* alunos = (Aluno*)malloc(2  *
sizeof(Aluno));
    leVetorAluno(alunos,  2);
    imprimeVetorAluno(alunos,  2);
    free (alunos);
    return 0;
}
```

Exercício 4

Crie uma estrutura ponto contendo os campos x e y, coordenadas de um ponto dadas por valores reais. Declare as variáveis p1 e p2 (dois pontos), preencha os campos para cada variável, via teclado, e imprima a distancia entre p1 e p2.

Nota: A distancia entre dois pontos é dada pela fórmula seguinte: d = $\sqrt{((x2 - x1)2 + (y2 - y1)2)}$.

Considere, p1(x1 , y1) e p2(x2 , y2).

Exercício 4 - Resposta

```
typedef struct ponto {
    int x,y;
}p1,p2;

main ()
{
    float distancia;
    printf ("\nDigite (X,Y) de P1.\n");
    scanf ("%d %d",&p1.x,&p1.y);
    printf ("\nDigite (X,Y) de P2.\n");
    scanf ("%d %d",&p2.x,&p2.y);
    distancia=sqrt(pow(p2.x-p1.x,2)+pow(p2.y-p1.y,2));
    printf ("\nA distância entre P1 e P2 e:%.1f.\n\n",distancia);
    system("pause");
    return 0;
}
```