Roteiro 3 - Laboratório de Programação 2

1.1 - Um Tipo Abstrato de Dados (TAD) é como um modelo matemático, com um conjunto de operações definidas sobre o modelo. Por exemplo, o conjunto dos números inteiros, acompanhado das operações: soma, subtração, multiplicação e divisão é um tipo abstrato de dados. A principal vantagem ao se utilizar um TAD é o encapsulamento, uma vez que a definição do tipo de dado e todas as operações definidas sobre ele ficam localizadas em uma única seção do programa.

1.2 -

```c
#include <stdio.h>

typedef struct{
    double lado;
} Cubo;

double calcularAreaCubo(Cubo *cubo){
    return cubo->lado * cubo->lado;
}

double calcularVolumeCubo(Cubo *cubo){
    return cubo->lado * cubo->lado * cubo->lado;
}

double tamanhoLadosCubo(Cubo *cubo){
    return cubo->lado;
}

int main(){
    Cubo cubo;
    cubo.lado = 5.0;

    printf("Tamanho dos Lados do Cubo: %.2f\n", tamanhoLadosCubo(&cubo));
    printf("Area do Cubo: %.2f\n", calcularAreaCubo(&cubo));
    printf("Volume do Cubo: %.2f\n", calcularVolumeCubo(&cubo));

    return 0;
}
```

Saída:

```
PS C:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 3\output> cd 'c:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 3\output'
PS C:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 3\output> & .\'exercicio2.exe'
Tamanho dos Lados do Cubo: 5.00
Area do Cubo: 25.00
Volume do Cubo: 125.00
```

1.3 -

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

#define MAX_NUMEROS 10

typedef struct{
    int vetor[MAX_NUMEROS];
    int tamanhoConjunto;
} ConjuntoInteiros;

void criarConjuntoVazio(ConjuntoInteiros *conjunto){
    conjunto->tamanhoConjunto = 0;
}

ConjuntoInteiros uniaoConjuntos(ConjuntoInteiros *conjunto1, ConjuntoInteiros *conjunto2){
    ConjuntoInteiros conjunto3;
    criarConjuntoVazio(&conjunto3);

    for (int i=0; i < conjunto1->tamanhoConjunto; i++){
        inserirElementoNoConjunto(&conjunto3, conjunto1->vetor[i]);
    }

    for (int i=0; i < conjunto2->tamanhoConjunto; i++){
        inserirElementoNoConjunto(&conjunto3, conjunto2->vetor[i]);
    }

    return conjunto3;
}

void inserirElementoNoConjunto(ConjuntoInteiros *conjunto, int elemento){
    if (conjunto->tamanhoConjunto < MAX_NUMEROS){
        conjunto->vetor[conjunto->tamanhoConjunto] = elemento;
        conjunto->tamanhoConjunto++;
    } else {
        printf ("Conjunto cheio.\n");
    }
}
```

```
void    removerElementoDoConjunto(ConjuntoInteiros    *conjunto,    int
elemento){
    for (int i=0; i< conjunto->tamanhoConjunto; i++){
        if (conjunto->vetor[i] == elemento){
            for (int j=i; i < conjunto->tamanhoConjunto; j++)
                conjunto->vetor[j] = conjunto->vetor[j+1];
            conjunto->tamanhoConjunto--;
        }

        return;
    }
}

ConjuntoInteiros interseccaoEntreConjuntos(ConjuntoInteiros *conjunto1,
ConjuntoInteiros *conjunto2){
    ConjuntoInteiros conjuntoInterseccao;
    criarConjuntoVazio(&conjuntoInterseccao);

    for (int i=0; i < conjunto1->tamanhoConjunto; i++){
        for (int j=0; i < conjunto2->tamanhoConjunto; j++){
            if (conjunto1->vetor[i] == conjunto2->vetor[j]){
                        inserirElementoNoConjunto(&conjuntoInterseccao,
conjunto1->vetor[i]);
                break;
            }
        }
    }

    return conjuntoInterseccao;
}

ConjuntoInteiros  diferencaEntreConjuntos(ConjuntoInteiros  *conjunto1,
ConjuntoInteiros *conjunto2){
    ConjuntoInteiros conjuntoDiferenca;
    criarConjuntoVazio(&conjuntoDiferenca);

    for (int i=0; i < conjunto1->tamanhoConjunto; i++){
         if (procurarNumeroNoConjunto(conjunto2, conjunto1->vetor[i]) ==
-1)
                          inserirElementoNoConjunto(&conjuntoDiferenca,
conjunto1->vetor[i]);
    }
```

```
    return conjuntoDiferenca;
}

int procurarNumeroNoConjunto(ConjuntoInteiros *conjunto, int elemento){
    for (int i=0; i < conjunto->tamanhoConjunto; i++){
        if (conjunto->vetor[i] == elemento)
            return i;
    }

    return -1;
}

int encontrarMaiorValor(ConjuntoInteiros *conjunto){
    if (conjunto->tamanhoConjunto == 0){
        printf ("Conjunto vazio.\n");
        return -1;
    }

    int maiorValor = conjunto->vetor[0];
    for (int i=0 ; i < conjunto->tamanhoConjunto; i++){
        if (maiorValor < conjunto->vetor[i])
            maiorValor = conjunto->vetor[i];
    }

    return maiorValor;
}

int encontrarMenorValor(ConjuntoInteiros *conjunto){
    if (conjunto->tamanhoConjunto == 0){
        printf ("Conjunto vazio.\n");
        return -1;
    }

    int menorValor = conjunto->vetor[0];
    for (int i = 1; i < conjunto->tamanhoConjunto; i++){
        if (menorValor > conjunto->vetor[i])
            menorValor = conjunto->vetor[i];
    }

    return menorValor;
}
```

```
int   conjuntosIguais(ConjuntoInteiros   *conjunto1,   ConjuntoInteiros
*conjunto2){
    int diferentes = 1;
    for (int i=0; i < conjunto1->tamanhoConjunto; i++){
        if (procurarNumeroNoConjunto(conjunto2, conjunto1->vetor[i]) ==
-1){
            diferentes = 0;
            break;
        }
    }

    return diferentes;
}

int tamanhoDoConjunto(ConjuntoInteiros *conjunto){
    return conjunto->tamanhoConjunto;
}

int conjuntoEstaVazio(ConjuntoInteiros *conjunto){
    return (tamanhoDoConjunto(conjunto) == 0);
}

int main(){
          ConjuntoInteiros   conjunto1,   conjunto2,   conjuntoDiferenca,
conjuntoInterseccao, conjuntoUniao;
    int maior2, menor2, tamanho1;

    criarConjuntoVazio(&conjunto1);
    inserirElementoNoConjunto(&conjunto1, 5);
    inserirElementoNoConjunto(&conjunto1, 10);
    inserirElementoNoConjunto(&conjunto1, 20);

    criarConjuntoVazio(&conjunto2);;
    inserirElementoNoConjunto(&conjunto2, 10);
    inserirElementoNoConjunto(&conjunto2, 20);
    inserirElementoNoConjunto(&conjunto2, 30);

    conjuntoUniao = uniaoConjuntos(&conjunto1, &conjunto2);
    printf ("Uniao dos Conjuntos: \n");
    for (int i=0; i < conjuntoUniao.tamanhoConjunto; i++)
        printf ("%d ", conjuntoUniao.vetor[i]);
    printf ("\n");
```

```
          conjuntoInterseccao   =   interseccaoEntreConjuntos(&conjunto1,
&conjunto2);
    printf ("Interseccao entre Conjuntos 1 e 2: \n");
    for (int i=0; i < conjuntoInterseccao.tamanhoConjunto; i++)
        printf ("%d ", conjuntoInterseccao.vetor[i]);
    printf ("\n");

            conjuntoDiferenca   =   diferencaEntreConjuntos(&conjunto1,
&conjunto2);
    printf ("Diferenca entre Conjuntos 1 e 2: \n");
    for (int i=0; i < conjuntoDiferenca.tamanhoConjunto; i++)
        printf ("%d ", conjuntoDiferenca.vetor[i]);
    printf ("\n");

    maior2 = encontrarMaiorValor(&conjunto2);
    menor2 = encontrarMenorValor(&conjunto2);
    tamanho1 = tamanhoDoConjunto(&conjunto1);
    printf("Maior valor no conjunto2: %d\n", maior2);
    printf("Menor valor no conjunto2: %d\n", menor2);
     printf("Os conjuntos sao iguais? %s\n", conjuntosIguais(&conjunto1,
&conjunto2) ? "Sim" : "Nao");
    printf("Tamanho do conjunto1: %d\n", tamanho1);
                  printf("O     conjunto1     esta     vazio?     %s\n",
conjuntoEstaVazio(&conjunto1) ? "Sim" : "Nao");

}
```

Saída:

```
PS C:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 3\output> cd 'c:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 3\output'
PS C:\Users\USER\OneDrive\Área de Trabalho\LabProg2\Lista 3\output> & .\'exercicio1.exe'
Uniao dos Conjuntos:
5 10 20 10 20 30
Interseccao entre Conjuntos 1 e 2:
5 10 20
Diferenca entre Conjuntos 1 e 2:
5
Maior valor no conjunto2: 30
Menor valor no conjunto2: 10
Os conjuntos sao iguais? Nao
Tamanho do conjunto1: 3
O conjunto1 esta vazio? Nao
```