

Nome: Messias Feres Curi Melo

Matrícula: 2022003764

1 Tipos Abstratos de Dados

1.1 TAD é uma **estrutura de dados** que possui detalhes 'escondidos' durante a implementação do software, facilitando para se programar, reutilizar códigos, fazer manutenção, para segurança do código, entre outras vantagens. No geral ele faz uma abstração dos dados e modulariza o projeto.

1.2

main.c

```
C: > Users > Messi > OneDrive > Área de Trabalho > lista3 > C main.c > main()

1  #include "cubo.h"
2  #include <stdio.h>
3  #include <stdlib.h>
4
5  int main(){
6      int n;
7      Cubo c;
8      printf("Número de lados : ");
9      scanf("%d", &n);
10     criarCubo(&c, n);
11     printf("-> Valor do lado : %d\n", medidaCubo(c.lado));
12     printf("-> Valor da área : %d\n", areaCubo(c.lado));
13     printf("-> Valor do volume : %d\n", volumeCubo(c.lado));
14     return 0;
15 }
```

cubo.c

```
C: > Users > Messi > OneDrive > Área de Trabalho > lista3 > C cubo.c > medidaCubo(int)

1  #include "cubo.h"
2  #include <stdio.h>
3  #include <stdlib.h>
4
5  void criarCubo(Cubo *c, int n){
6      c->lado = n;
7  }
8
9  int medidaCubo(int n){
10     return n;
11 }
12
13 int areaCubo(int n){
14     return (n*n)*6;
15 }
16
17 int volumeCubo(int n){
18     return (n*n*n);
19 }
```

cubo.h

```
C: > Users > Messi > OneDrive > Área de Trabalho > lista3 > C cubo.h > areaCubo(int)

1  #ifndef CUBO_H
2  #define CUBO_H
3
4  typedef struct{
5      int lado;
6  }Cubo;
7
8  void criarCubo(Cubo *c, int n);
9  int medidaCubo(int n);
10 int areaCubo(int n);
11 int volumeCubo(int n);
12
13 #endif
```

Terminal:

```
messiasfcn@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/lista3$ gcc main.c cubo.c cubo.h -o tp2
messiasfcn@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/lista3$ ./tp2
Número de lados : 10
-> Valor do lado : 10
-> Valor da área : 600
-> Valor do volume : 1000
messiasfcn@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/lista3$
```

1.3

main.c

```
C:\Users> Messi > OneDrive > Área de Trabalho > lista3_3 > C main.c > main()
1  #include "conjunto.h"
2  #include <stdio.h>
3  #include <stdlib.h>
4  #include <stdbool.h>
5
6  int main(){
7      int escolha, elemento, conjuntoNumero, conjuntoNumero2, resultado;
8      listadeConjuntos c;
9      c.tamanho = 0;
10     c.conjuntos = NULL;
11
12     printf("--- MENU ---\n");
13     printf("1 - Criar um conjunto vazio\n");
14     printf("2 - União de dois conjuntos\n");
15     printf("3 - Inserir elemento\n");
16     printf("4 - Remover elemento\n");
17     printf("5 - Intersecção entre dois conjuntos\n");
18     printf("6 - Diferença de dois conjuntos\n");
19     printf("7 - Número pertence ao conjunto\n");
20     printf("8 - Menor valor do conjunto\n");
21     printf("9 - Maior valor do conjunto\n");
22     printf("10 - Testar se dois conjuntos são iguais\n");
23     printf("11 - Tamanho do conjunto\n");
24     printf("12 - Testar se o conjunto é vazio\n");
25     printf("0 - Finalizar\n");
26
27     while (true) {
28         printf("Valor de escolha : ");
29         scanf("%d", &escolha);
30
31         switch (escolha) {
32             case 1:
33                 criarConjunto(&c);
34                 printf("Novo conjunto criado\n");
35                 break;
36             case 2:
37                 printf("Insira o número do primeiro conjunto : ");
38                 scanf("%d", &conjuntoNumero);
39                 printf("Insira o número do segundo conjunto : ");
40                 scanf("%d", &conjuntoNumero2);
41                 Conjunto uniao = uniaoConjuntos(c.conjuntos[conjuntoNumero - 1], c.conjuntos[conjuntoNumero2 - 1]);
42                 printf("União dos conjuntos é : ");
43                 for(int i = 0; i<uniao.quantidadeElementos; i++){
44                     printf("%d ", uniao.elementos[i]);
45                 }
46                 limpar(&uniao);
47                 printf("\n");
48                 break;
49             case 3:
50                 printf("Insira qual o número do conjunto : ");
51                 scanf("%d", &conjuntoNumero);
52                 printf("Insira o número a ser adicionado : ");
53                 scanf("%d", &elemento);
54                 inserirElemento(&c.conjuntos[conjuntoNumero-1], elemento);
55                 break;
56             case 4:
57                 printf("Insira qual o número do conjunto : ");
58                 scanf("%d", &conjuntoNumero);
59                 printf("Insira o número a ser removido : ");
60                 scanf("%d", &elemento);
61                 removerElemento(&c.conjuntos[conjuntoNumero-1], elemento);
62                 break;
63             case 5:
64                 printf("Insira o número do primeiro conjunto : ");
65                 scanf("%d", &conjuntoNumero);
66                 printf("Insira o número do segundo conjunto : ");
67                 scanf("%d", &conjuntoNumero2);
68                 Conjunto intersecao = intersecaoConjuntos(c.conjuntos[conjuntoNumero - 1], c.conjuntos[conjuntoNumero2 - 1]);
69                 printf("Intersecção dos conjuntos é : ");
70                 for(int i = 0; i<intersecao.quantidadeElementos; i++){
71                     printf("%d ", intersecao.elementos[i]);
72                 }
73                 limpar(&intersecao);
74                 printf("\n");
75                 break;
76             case 6:
```

```

77     printf("Insira o número do primeiro conjunto : ");
78     scanf("%d", &conjuntoNumero);
79     printf("Insira o número do segundo conjunto : ");
80     scanf("%d", &conjuntoNumero2);
81     Conjunto diferenca = diferencaConjuntos(c.conjuntos[conjuntoNumero - 1], c.conjuntos[conjuntoNumero2 - 1]);
82     printf("Diferença dos conjuntos é : ");
83     for(int i = 0; i < diferenca.quantidadeElementos; i++){
84         printf("%d ", diferenca.elementos[i]);
85     }
86     limpar(&diferenca);
87     printf("\n");
88     break;
89 case 7:
90     printf("Insira qual o número do conjunto : ");
91     scanf("%d", &conjuntoNumero);
92     printf("Insira o número a ser verificado : ");
93     scanf("%d", &elemento);
94     resultado = pertence(c.conjuntos[conjuntoNumero-1], elemento);
95     if(resultado){
96         printf("Pertence\n");
97     }else{
98         printf("Não pertence\n");
99     }
100    break;
101 case 8:
102     printf("Insira qual o número do conjunto : ");
103     scanf("%d", &conjuntoNumero);
104     resultado = menorValor(c.conjuntos[conjuntoNumero-1]);
105     printf("O menor elemento é : %d\n", resultado);
106     break;
107 case 9:
108     printf("Insira qual o número do conjunto : ");
109     scanf("%d", &conjuntoNumero);
110     resultado = maiorValor(c.conjuntos[conjuntoNumero-1]);
111     printf("O maior elemento é : %d\n", resultado);
112     break;
113 case 10:
114     printf("Insira o número do primeiro conjunto : ");
115     scanf("%d", &conjuntoNumero);
116     printf("Insira o número do segundo conjunto : ");
117     scanf("%d", &conjuntoNumero2);
118     resultado = saoIguais(c.conjuntos[conjuntoNumero-1], c.conjuntos[conjuntoNumero2-1]);
119     if(resultado){
120         printf("São iguais\n");
121     }else{
122         printf("Não são iguais\n");
123     }
124     break;
125 case 11:
126     printf("Insira qual o número do conjunto : ");
127     scanf("%d", &conjuntoNumero);
128     resultado = tamanho(c.conjuntos[conjuntoNumero-1]);
129     printf("O tamanho é : %d\n", resultado);
130     break;
131 case 12:
132     printf("Insira qual o número do conjunto : ");
133     scanf("%d", &conjuntoNumero);
134     resultado = eVazio(c.conjuntos[conjuntoNumero-1]);
135     if(resultado){
136         printf("É vazio\n");
137     }else{
138         printf("Não é vazio\n");
139     }
140     break;
141 case 0:
142     break;
143 default:
144     printf("Escolha inválida\n");
145     break;
146 }
147 if(escolha == 0){
148     for(int i = 0; i < c.tamanho; i++){
149         limpar(&c.conjuntos[i]);
150     }
151     break;
152 }
153 }
154 return 0;
155 }
156 }

```

conjunto.c

```
C:\> Users > Messi > OneDrive > Área de Trabalho > lista3_3 > C conjunto.c > criarConjunto(listadeConjuntos *)

1  #include "conjunto.h"
2  #include <stdio.h>
3  #include <stdlib.h>
4  #include <stdbool.h>
5
6  Conjunto criar_conjunto_vazio() {
7      Conjunto n;
8      n.elementos = NULL;
9      n.quantidadeElementos = 0;
10     return n;
11 }
12
13 void criarConjunto(listadeConjuntos *c){
14     if(c->tamanho == 0) {
15         c->tamanho = 1;
16         c->conjuntos = (Conjunto*)malloc(sizeof(Conjunto));
17         c->conjuntos[0] = criar_conjunto_vazio();
18     }else {
19         c->tamanho++;
20         int nt = sizeof(listadeConjuntos);
21         c->conjuntos = (Conjunto*)realloc(c->conjuntos, c->tamanho*sizeof(Conjunto));
22         c->conjuntos[c->tamanho-1] = criar_conjunto_vazio();
23     }
24 }
25
26 Conjunto uniaoConjuntos(Conjunto c1, Conjunto c2) {
27     Conjunto uniao = criar_conjunto_vazio();
28     for (int i = 0; i < c1.quantidadeElementos; i++) {
29         inserirElemento(&uniao, c1.elementos[i]);
30     }
31     for (int i = 0; i < c2.quantidadeElementos; i++) {
32         if (!pertence(uniao, c2.elementos[i])) {
33             inserirElemento(&uniao, c2.elementos[i]);
34         }
35     }
36     return uniao;
37 }
38
39 void inserirElemento(Conjunto *c, int elemento) {
40     for (int i = 0; i < c->quantidadeElementos; i++) {
41         if (c->elementos[i] == elemento) {
42             return;
43         }
44     }
45     c->quantidadeElementos++;
46     c->elementos = (int *)realloc(c->elementos, c->quantidadeElementos * sizeof(int));
47     c->elementos[c->quantidadeElementos - 1] = elemento;
48 }
49
50 void removerElemento(Conjunto *c, int elemento) {
51     for (int i = 0; i < c->quantidadeElementos; i++) {
52         if (c->elementos[i] == elemento) {
53             for (int j = i; j < c->quantidadeElementos - 1; j++) {
54                 c->elementos[j] = c->elementos[j + 1];
55             }
56             c->quantidadeElementos--;
57             c->elementos = (int *)realloc(c->elementos, c->quantidadeElementos * sizeof(int));
58             return;
59         }
60     }
61 }
62
63 Conjunto intersecaoConjuntos(Conjunto c1, Conjunto c2) {
```

```

64     Conjunto intersecao = criar_conjunto_vazio();
65     for (int i = 0; i < c1.quantidadeElementos; i++) {
66         if (pertence(c2, c1.elementos[i])) {
67             inserirElemento(&intersecao, c1.elementos[i]);
68         }
69     }
70     return intersecao;
71 }
72
73 Conjunto diferencaConjuntos(Conjunto c1, Conjunto c2) {
74     Conjunto diferenca = criar_conjunto_vazio();
75     for (int i = 0; i < c1.quantidadeElementos; i++) {
76         if (!pertence(c2, c1.elementos[i])) {
77             inserirElemento(&diferenca, c1.elementos[i]);
78         }
79     }
80     return diferenca;
81 }
82
83 int pertence(Conjunto c, int numero) {
84     for (int i = 0; i < c.quantidadeElementos; i++) {
85         if (c.elementos[i] == numero) {
86             return 1;
87         }
88     }
89     return 0;
90 }
91
92 int menorValor(Conjunto c) {
93     if (c.quantidadeElementos == 0) {
94         return -1;
95     }
96     int menor = c.elementos[0];
97     for (int i = 1; i < c.quantidadeElementos; i++) {
98         if (c.elementos[i] < menor) {
99             menor = c.elementos[i];
100         }
101     }
102     return menor;
103 }
104
105 int maiorValor(Conjunto c) {
106     if (c.quantidadeElementos == 0) {
107         return -1;
108     }
109     int maior = c.elementos[0];
110     for (int i = 1; i < c.quantidadeElementos; i++) {
111         if (c.elementos[i] > maior) {
112             maior = c.elementos[i];
113         }
114     }
115     return maior;
116 }
117
118 int saoIguais(Conjunto c1, Conjunto c2) {
119     if (c1.quantidadeElementos != c2.quantidadeElementos) {
120         return false;
121     }

```

```

122     for (int i = 0; i < c1.quantidadeElementos; i++) {
123         if (!pertence(c2, c1.elementos[i])) {
124             return false;
125         }
126     }
127     return true;
128 }
129
130 int tamanho(Conjunto c) {
131     if(c.elementos == NULL || c.quantidadeElementos == 0){
132         return 0;
133     }else{
134         return c.quantidadeElementos;
135     }
136 }
137
138 int eVazio(Conjunto c) {
139     if(c.elementos == NULL || c.quantidadeElementos == 0){
140         return true;
141     }else{
142         return false;
143     }
144 }
145
146 void limpar(Conjunto *c) {
147     free(c->elementos);
148     c->elementos = NULL;
149     c->quantidadeElementos = 0;
150 }
151

```

conjunto.h

```

C: > Users > Messi > OneDrive > Área de Trabalho > lista3_3 > C conjunto.h > criar_conjunto_vazio()
1  #ifndef CONJUNTO_H
2  #define CONJUNTO_H
3
4  typedef struct{
5      int *elementos;
6      int quantidadeElementos;
7  }Conjunto;
8
9  typedef struct{
10     Conjunto *conjuntos;
11     int tamanho;
12 }listadeConjuntos;
13
14 Conjunto criar_conjunto_vazio();
15 void criarConjunto(listadeConjuntos *c);
16 Conjunto uniaoConjuntos(Conjunto c1, Conjunto c2);
17 void inserirElemento(Conjunto *c, int elemento);
18 void removerElemento(Conjunto *c, int elemento);
19 Conjunto intersecaoConjuntos(Conjunto c1, Conjunto c2);
20 Conjunto diferencaConjuntos(Conjunto c1, Conjunto c2);
21 int pertence(Conjunto c, int numero);
22 int menorValor(Conjunto c);
23 int maiorValor(Conjunto c);
24 int saoIguais(Conjunto c1, Conjunto c2);
25 int tamanho(Conjunto c);
26 int eVazio(Conjunto c);
27 void limpar(Conjunto *c);
28
29 #endif

```

Terminal:

```
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/lista3_3$ gcc main.c conjunto.c conjunto.h -o tp3
^[[Amessiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/lista3_3$ ./tp3
--- MENU ---
1 - Criar um conjunto vazio
2 - União de dois conjuntos
3 - Inserir elemento
4 - Remover elemento
5 - Intersecção entre dois conjuntos
6 - Diferença de dois conjuntos
7 - Número pertence ao conjunto
8 - Menor valor do conjunto
9 - Maior valor do conjunto
10 - Testar se dois conjuntos são iguais
11 - Tamanho do conjunto
12 - Testar se o conjunto é vazio
0 - Finalizar
Valor de escolha : 1
Novo conjunto criado
Valor de escolha : 1
Novo conjunto criado
Valor de escolha : 3
Insira qual o número do conjunto : 1
Insira o número a ser adicionado : 2
Valor de escolha : 3
Insira qual o número do conjunto : 2
Insira o número a ser adicionado : 1
Valor de escolha : 3
Insira qual o número do conjunto : 1
Insira o número a ser adicionado : 23
Valor de escolha : 8
Insira qual o número do conjunto : 1
O menor elemento é : 2
Valor de escolha : 6
Insira o número do primeiro conjunto : 1
Insira o número do segundo conjunto : 2
Diferença dos conjuntos é : 2 23

Valor de escolha : 11
Insira qual o número do conjunto : 1
O tamanho é : 2
Valor de escolha : 12
Insira qual o número do conjunto : 2
Não é vazio
Valor de escolha : 7
Insira qual o número do conjunto : 1
Insira o número a ser verificado : 2
Pertence
Valor de escolha : 2
Insira o número do primeiro conjunto : 1
Insira o número do segundo conjunto : 2
União dos conjuntos é : 2 23 1
Valor de escolha : 0
messiasfcm@MessiasFCM:/mnt/c/Users/Messi/OneDrive/Área de Trabalho/lista3_3$
```