Algoritmos e Estruturas de Dados II

Aula II - TAD

Tipos Compostos Heterogêneos

- As Estruturas são entidades que representam tipos de dados e que permitem o agrupamento de variáveis de diversos tipos.

- ('Jose', '01/04/1980', 1.76)

Tipos Compostos Heterogêneos

- Alternativa aos tipos de dados simples, fornecidos pela linguagem:
  - Definição fica a cargo do programador, de acordo com o problema;
  - Permite que uma função retorne mais de um valor.

- Exemplo:
  - Matrícula, idade, coeficiente de rendimento e período

Tipos Compostos Heterogêneos

```
struct <nome da estrutura>{
<tipo do atributo 1>  <nome do atributo 1>;
<tipo do atributo 2>  <nome do atributo 2>;
...
<tipo do atributo n>  <nome do atributo n>;
};
```

```
1  struct tEstudante{
2      int idade;
3      int matricula;
4      float coeficiente;
5      int periodo;
6  };
```

Tipos Compostos Heterogêneos

Tipos Compostos Heterogêneos

- Acesso seletivo aos elementos da estrutura

```
<nome da variável>.<nome do atributo>
```

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  struct tEstudante{
5      int idade;
6      int matricula;
7      float coeficiente;
8      int periodo;
9  };
10
11 main(){
12     struct tEstudante aluno;
13     printf("Digite os dados do estudante");
14     scanf("%d %d %f %d",&aluno.idade,&aluno.matricula,&aluno.coeficiente,&aluno.
           periodo);
15 }
```

Tipos Compostos Heterogêneos

- Acesso integral à estrutura

```
<nome da variável 1> = <nome da variável 2>
```

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  struct tEstudante{
5     int idade, matricula, periodo;
6     float coeficiente;
7  };
8
9  main(){
10    struct tEstudante aluno,outro;
11    printf("Digite os dados do estudante");
12    scanf("%d %d %f %d",&aluno.idade,&aluno.matricula,&aluno.coeficiente,&aluno.
          periodo);
13    outro = aluno;
14 }
```

Tipos Compostos Heterogêneos

- Redefinição de tipos

```
1  typedef struct tEstudante tEstudante;
2
3  typedef float distancia;
```

Tipos Compostos Heterogêneos

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  struct tEstudante{
5     int idade;
6     int matricula;
7     float coeficiente;
8     int periodo;
9  };
10
11 typedef struct tEstudante tEstudante;
12
13 main(){
14    tEstudante aluno,outro;
15    printf("Digite os dados do estudante");
16    scanf("%d %d %f %d",&aluno.idade,&aluno.matricula,&aluno.coeficiente,&aluno.
          periodo);
17    outro=aluno;
18 }
```

Tipos Compostos Heterogêneos

- Faça um programa que leia dois pontos do plano cartesiano, calcule a distancia entre eles e imprima o resultado na tela.
  - Defina uma estrutura para o ponto
  - Defina um tipo para renomear a estrutura
  - Use uma função para cálculo da distancia

Tipos Abstratos de Dados (TAD)

- São novos tipos de dados implementados pelo programador, nos quais ele define tanto as estruturas de dados quanto as operações a elas aplicáveis

- Exemplo, suponha que desejamos definir um tipo abstrato de dados para data.

```
typedef struct data{
    int dia;
    int mes;
    int ano;
}tData;
```

Tipos Abstratos de Dados (TAD)

- Podemos imaginas várias operações que desejaríamos realizar com instâncias deste tipo de dado

- Inicializar data
  - a partir de valores passados como parâmetro.
  - a partir de valores lidos do teclado.
- Altera data
  - a partir de valores passados como parâmetro.
  - para dia seguinte
- Verificar se uma data está num ano bissexto
- Indicar a quantidade de dias do mês em questão.

Tipos Abstratos de Dados (TAD)

```
1  #include <stdio.h>
2
3  typedef struct data{
4     int dia;
5     int mes;
6     int ano;
7  }tData;
8
9  tData inicializarValores(int d,int m,int a){
10    tData dt;
11
12    dt.dia=d;
13    dt.mes=m;
14    dt.ano=a;
15    return dt;
16 }
17
18 tData leData(){
19    tData d;
20
21    printf("Entre com a data");
22    scanf("%d %d %d",&d.dia,&d.mes,&d.ano);
23    return d;
24 }
25
```

Tipos Abstratos de Dados (TAD)

```
26 tData alteraData(int d,int m,int a){
27    tData dt;
28
29    dt.dia=d;
30    dt.mes=m;
31    dt.ano=a;
32    return dt;
33 }
34
35 int eBissexto(tData d){
36    if(d.ano%400==0){
37       return 1;
38    }else if(d.ano%100==0){
39       return 0;
40    }else if(d.ano%4==0){
41       return 1;
42    }else{
43       return 0;
44    }
45 }
46
```

Tipo Abstratos de Dados (TAD)

```
47  int diasNoMes(tData d){
48     if(d.mes==4||d.mes==6||d.mes==9||d.mes==11){
49        return 30;
50     }else{
51        if(d.mes=2){
52           if(eBissexto(d)){
53              return 29;
54           }else{
55              return 28;
56           }
57        }else{
58           return 31;
59        }
60     }
61  }
62
63  tData diaSeguinte(tData d){
64     if(d.dia<diasNoMes(d)){
65        d.dia++;
66     }else{
67        d.dia=1;
68        if (d.mes<12){
69           d.mes++;
70        }else{
71           d.mes=1;
72           d.ano++;
73        }
74     }
75     return d;
76  }
```

Tipos Abstratos de Dados (TAD)

- Ou seja, temos
  - um tipo composto, ou seja, uma estrutura que agrupa variáveis correspondente a dia mês e ano
  - uma série de operações pré-definidas para tal tipo

- Vantagens
  - Manutenibilidade
  - Reusabilidade
  - Abstração
  - Ocultamento
  - Integridade

Tipos Abstratos de Dados (TAD)

- Problema do uso de TAD em C:
  - A linguagem C não implementa de fato TAD, apenas simula
  - O programador-usuário do TAD tem acesso a alterar diretamente os valores sem usar as operações pré-definidas.

Exemplo de uso do TAD

```
1  #include <stdio.h>
2
3  // A definição do TAD, bem como suas operações, está no Exemplo
4
5  main(){
6     tData data;
7     int anoBissexto;
8     int nDias;
9
10    data=leData();
11    anoBissexto=eBissexto(data);
12
13    if(anoBissexto == 1){
14       printf("Ano Bissexto");
15    }else{
16       printf("Ano não Bissexto");
17    }
18
19    nDias=diasNoMes(data);
20    printf("Número de dias no mês:",nDias);
21 }
```

Tipos de Operação de TAD

- Construtoras
  - São operações que realizam a inicialização dos valores

```
1  tEstudante inicializar(){
2     tEstudante novo;
3
4     novo.idade=0;
5     novo.matricula=0;
6     novo.coeficiente=0;
7     novo.periodo=0;
8     return novo;
9  }
10
11 tEstudante inicializarValores(int idade,int matricula,float coeficiente, int
       periodo){
12    tEstudante novo;
13
14    novo.idade=idade;
15    novo.matricula=matricula;
16    novo.coeficiente=coeficiente;
17    novo.periodo=periodo;
18    return novo;
19 }
```

Tipos de Operação de TAD

- Analisadoras ou consultoras
  - Analisam o conteúdo de um TAD e retornam uma propriedade

```
1  int bomAluno(tEstudante aluno){
2      if (aluno.coeficiente>7.0){
3          return 1;
4          }else{
5              return 0;
6          }
7  }
```

Tipos de Operação de TAD

- Modificadoras ou atualizadoras
  - Realizam a alteração de atributos de umTAD

```
1  tEstudante alteraIdade(tEstudante aluno,int novaIdade){
2      aluno.idade=novaIdade;
3      return aluno;
4  }
```

Tipos de Operação de TAD

- Produtoras
  - Comparam dados de um TAD e produzem nova informação.

```
1  int maiorIdade(tEstudante est1,tEstudante est2){
2      if (est1.idade>est2.idade){
3          return est1.idade;
4      }
5      return est2.idade;
6  }
```

Tipos de Operação de TAD

- Destrutoras
  - São utilizadas para liberar recurso de memória ocupados por um TAD. Serão vistas mais adiante neste curso

Tipos de TAD

- TADs de domínio:
  - Definem tipos de dados diretamente relacionados ao domínio do problema

- TADs implementacionais
  - Tem relação direta com questões implementacionais, como listas, árvores, grafos e filas. Estes também serão vistos mais adiante neste curso.

Exercício

- Transforme o tipo de dados composto construído para um ponto no plano cartesiano em umTAD
  - Do que precisamos?
    - Uma estrutura que agrupa as variáveis
    - Uma série de operações pré-definidas
      - Construtoras
      - Analisadoras
      - Modificadoras
      - Produtoras