

Estruturas de dados

Tipos de dados abstratos (TDA)

Tipos de dados

Define o conjunto de valores que uma variável pode assumir

- **Tipos primitivos:**

Linguagem C: `char, int, float, double, void`

Ex: `int`

Domínio = $\{-2^{31} \text{ a } 2^{31}-1\}$

- **Tipos definidos pelo programador**

Ex: `typedef struct {
 int dia,mes,ano;
} Data;`

Utiliza mecanismos para a construção dos tipos (struct, typedef)

Tipos de dados abstratos

É um modelo matemático utilizado para especificar as características dos dados envolvidos no problema.

É independente de implementação

Conjunto de valores

+

Operações que podem ser aplicados sobre esses valores

Ex: inteiro

$\mathbb{Z} = \{ \dots, -3, -2, -1, 0, 1, 2, 3, \dots \}$

Operações = $\{ +, -, *, / \}$

Tipos de dados abstratos



(Definição do tipo e
Implementação das operações)

Em C:

int Faixa de valores: $\{-2^{31} \text{ a } 2^{31}-1\}$

Operações = $\{+, -, *, /, \%\}$

Tipos de dados abstratos

Em C:

int Faixa de valores: $\{-2^{31} \text{ a } 2^{31}-1\}$

Operações = $\{+, -, *, /, \%\}$

- Pode **não** existir uma implementação que represente o tipo completamente;
- **Separa** o conceito (definição) da implementação;
- Um mesmo tipo pode possuir **várias** implementações.

Tipos de dados abstratos

Como implementar em C?

- Mecanismos de agrupamento para definir a forma como os dados serão armazenados;
- Funções para definir as operações.

Tipos de dados abstratos

Exemplo: **Ponto**

(Armazena as coordenadas x,y de um ponto)

Operações:

```
void criaPonto(Ponto *pt, int x, int y);  
void escrevePonto(Ponto pt);  
void transladaPonto(Ponto *pt, int dx, int dy);  
float calcDistancia(Ponto pt1, Ponto pt2);
```

Tipos de dados abstratos

PROBLEMA:

Escreva um programa em C para ler as coordenadas de 2 pontos, calcular e escrever a distância entre eles, transladar o segundo ponto de 10 unidades na coordenada x e imprimir o ponto transladado.

Tipos de dados abstratos

```
int main() {  
    Ponto p1,p2;  
    int x,y;  
    float dist;  
  
    printf("Informa as coordenadas do ponto 1:");  
    scanf("%d %d", &x, &y);  
    criaPonto(&p1,x,y);  
  
    printf("Informa as coordenadas do ponto 2:");  
    scanf("%d %d", &x, &y);  
    criaPonto(&p2,x,y);  
  
    dist = calcDistancia(p1,p2);  
    printf("Distancia: %f\n",dist);  
  
    transladaPonto(&p2,10,0);  
    escrevePonto(p2);  
    return 0;  
}
```

Tipo Ponto

```
...    /* Os protótipos foram omitidos */  
  
typedef struct {  
    int x,y;  
} Ponto;  
  
void criaPonto(Ponto *pt,int x,int y) {  
    pt->x = x;  
    pt->y = y;  
}
```

Tipo Ponto

```
typedef struct {  
    int x,y;  
} Ponto;
```

```
void escrevePonto(Ponto pt) {  
    printf("( %d,%d) ", pt.x, pt.y);  
}
```

Tipo Ponto

```
typedef struct {  
    int x,y;  
} Ponto;
```

```
void transladaPonto(Ponto *pt, int dx, int dy) {  
    pt->x = pt->x + dx;  
    pt->y = pt->y + dy;  
}
```

Tipo Ponto

```
typedef struct {  
    int x,y;  
} Ponto;
```

```
float calcDistancia(Ponto pt1, Ponto pt2) {  
    float dist;  
  
    dist = sqrt( pow(pt1.x-pt2.x,2) +  
                 pow(pt1.y-pt2.y,2) );  
    return dist;  
}
```

