8

抽象数据类型
Abstract Data Type

郝家胜

hao@uestc.edu.cn

自动化工程学院

有理数的表示与使用

- 糟糕的设计

```
typedef struct {
    int numer;
    int denom;
} rational;

rational x, y, z;
x.numer = 1, x.denom = 2;
y.numer = 2, y.denom = 3;
add_complex(&z, x, y);
```

```
add_rational (rational *z,
  rational x, rational y,)
{
  z->numer = x.numer *
  y.denom + y.numer *
  x.denom;
  z->denom = x.denom *
  y.denom;

}
```

- 结构数据的使用不应依赖于数据结构的实现
  - 灵活性差
  - 复杂

数据抽象

- 为什么要使用复合数据？
  - 提升概念层次，提高模块性，增强表达能力
- 使用复合数据的程序，应该在抽象数据上操作
  - 最小允诺原则
    - 保留最少的必须特性，忽略一切其它细节，且不做任何假设
  - 数据的使用方式应与具体的表示与实现无关
- 将复合数据的使用，与该它怎样由更基本的数据构造起来的实现细节分隔开（黑箱抽象）
- 数据抽象屏障是控制复杂性的强有力工具

抽象数据类型 (ADT)

- ADT是定义了一组逻辑操作的数学模型
  - 由用户定义，用一组接口来操作抽象数据
  - 采用适当的数据结构来实现，并将与数据结构直接相关的操作封装起来，实现对外提供的接口
- ADT的两个重要特征
  - 数据抽象
    - 强调的是本质特征和外部接口，不对内部细节作任何假设
  - 数据封装
    - 将实体的外部特性和其内部实现细节分离
    - 对外部用户隐藏内部实现细节

基本数据类型与抽象数据类型

- 基本数据类型是高级程序语言对机器实现的抽象
  - 整数，字节数，字节序
- 抽象数据类型是问题域的数据类型
  - `typedef int bool;`
  - `typedef int BookId;`
- 抽象数据类型是对自定义的复合数据的抽象
- 数据结构是抽象数据类型的实现基础

有理数的抽象数据类型

- 数学模型
  - 有理数可由分子n和分母m来表示
  - n, m为整数，且m<>0
- 操作运算

```
new_rat(n, m)：给定分子和分母，构造一个有理数
del_rat(r)：释放有理数r所占的内存资源
rat_numer(r)：给定有理数r，返回其分子
rat_denom(r)：给定有理数r，返回其分母

add_rat(x, y)：给定有理数x和y，返回二者的和
sub_rat(x, y)：给定有理数x和y，返回二者的差
mul_rat(x, y)：给定有理数x和y，返回二者的积
div_rat(x, y)：给定有理数x和y，返回二者的商
```

有理数的使用

```
rational x, y, z;

x = new_rat(1, 2);
y = new_rat(2, 3);
z = add_rat(x, y);
```

```
add_rat(rational x, rational y)
{
   int numer, denom;
   numer = rat_numer(x) * rat_denom(y) +
           rat_numer(y) * rat_denom(x);
   denom = rat_denom(x) * rat_denom(y);
   return new_rat(numer, denom);
}
```

有理数的表示与实现 (1)

```
typedef struct ratiaonal_t {
  int numer;
  int denom;
} *rational;

rational new_rat(int n, int m)
{
  rational r = (rational)malloc(
               sizeof(struct rational_t));
  r->numer = n;
  r->denom = m;
  return r;
}
```

```
int rat_numer(rational x)
{
  return x->numer;
}
```

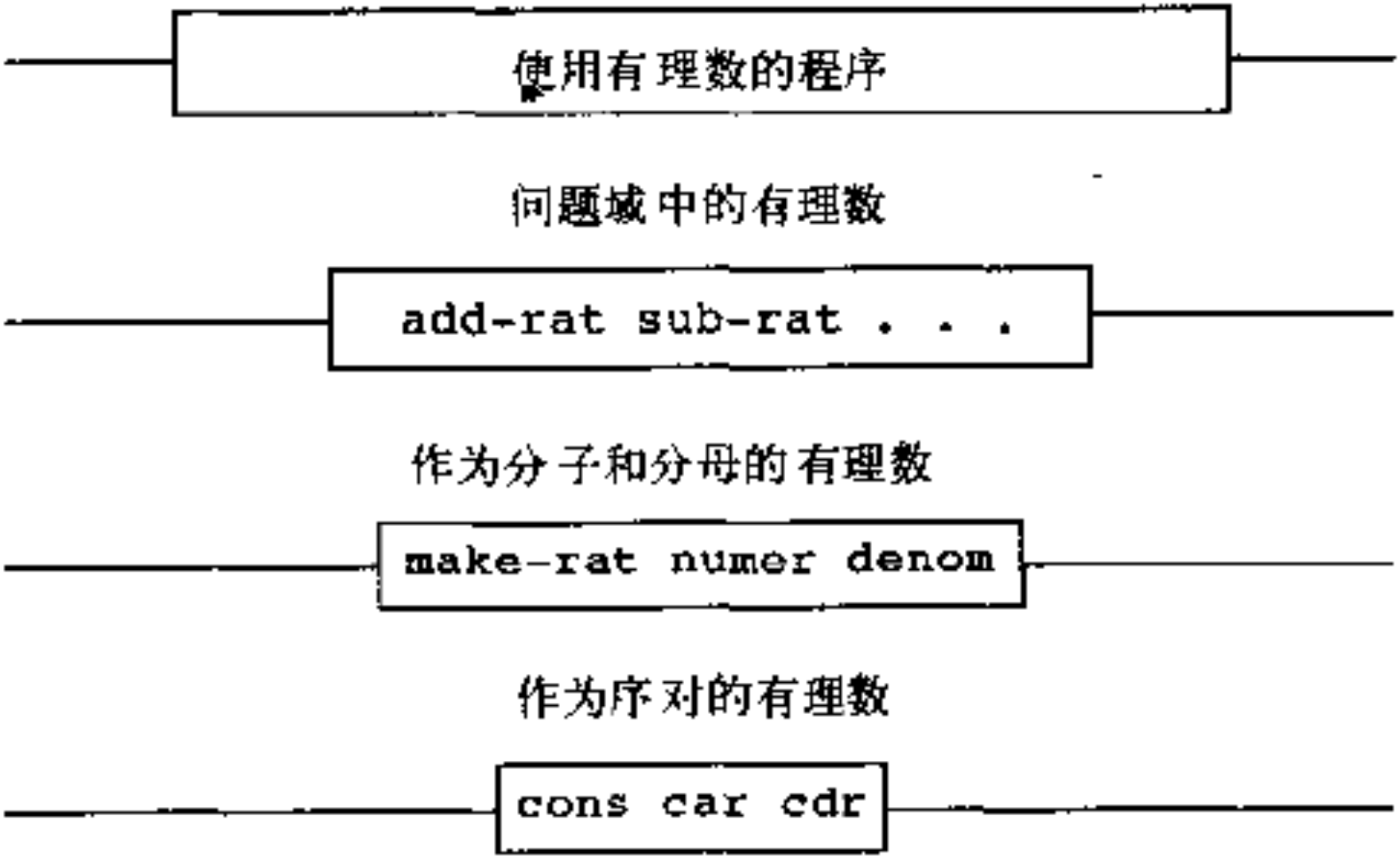
有理数的表示与实现（2)

```
typedef struct ratiaonal_t {
  int data[2];
} *rational;

rational new_rat(int n, int m)
{
  rational r = (rational)malloc(
               sizeof(struct rational_t));
  r->data[0] = n;
  r->data[1] = m;
  return r;
}
```

```
int rat_numer(rational x)
{
  return x->data[0];
}
```

数据的抽象屏蔽


使用有理数的程序
问题域中的有理数
add-rat sub-rat . . .
作为分子和分母的有理数
make-rat numer denom
作为序对的有理数
cons car cdr

数据结构与抽象数据类型

- 抽象数据类型是根据求解问题的需要，对问题域数据进行抽象所得的自定义的数据类型
  - 数学模型
  - 基于这个模型的一组操作方法
  - 采用适当的数据结构来实现
- 数据结构是抽象数据类型的内部表示和实现方式
- 抽象数据类型的好处
  - 从使用者的角度
  - 从实现者的角度
- 抽象数据类型为复合数据的使用提供了抽象方法，数据结构为复合数据的表示提供了实现方法

小结

- 抽象数据类型
- 数据结构与抽象数据类型的关系