抽象数据类型、实现及其应用

乔海燕
中山大学数据科学与计算机学院

2017 年 9 月 22 日

摘　要

以栈为例，介绍抽象数据类型的概念、实现及其应用。

§1　抽象数据类型的概念

在处理数据时，常常遇到对某类数据进行某些典型的操作。例如，对于整数需要进行加减乘除运算，对于字符串需要进行取查找子串、取字串和替换字串等操作。再比如，在许多场合需要对数据“后进先出”的操作，因此需要借助这样的装置实现数据处理。这种数据及其典型操作的数学模型称为**抽象数据类型**(Abstract Data Type)，简称ADT。例如，整数及其运算构成一个ADT；字符串及其常用操作构成一个ADT；在一个对象序列的同一端添加对象或者删除对象的模型构成一个ADT，及我们熟悉的栈。

这些ADT在程序设计语言的实现，为表示和处理数据提供了极大的方便。例如，在C++中int 及其附带的运算是整数ADT的实现，标准库STL中string是串ADT 的实现，stack 是栈的实现。程序员可以使用这些ADT表示数据和处理数据。

栈的定义

数据对象：同类型元素的线性序列$S = (a_1, a_2, \cdots, a_n)$。这里约定$a_1$为栈底，$a_n$为栈顶。

基本操作：

构造空栈initStack(&S)：S是一个空栈，即$S = ()$。

返回栈顶元素top(S)：如果当前栈$S == (a_1, a_2, \cdots, a_n)$,则返回栈顶$a_n$。

入栈push(&S, e)：将元素e插入栈顶，即如果原栈$S = (a_1, a_2, \cdots, a_n)$，则操作push($S$, e)的结果是e成为新的栈顶，即$S = (a_1, a_2, \cdots, a_n, e)$。

出栈pop(&S)：如果原栈不空$S = (a_1, a_2, \cdots, a_{n-1}, a_n)$，则操作pop($S$) 的结果是新栈为$S = (a_1, a_2, \cdots, a_{n-1})$。

检查栈是否空empty(S)：如果栈S空，则返回true，否则返回false。

§2 抽象数据类型栈的class定义和实现

栈的class定义

假定T是栈的元素类型，栈的class定义如下：

```
class Stack {
  public:
   //  Standard Stack methods
   Stack();
   bool empty() const;
   void push(const T &item);
   void pop();
   T top() const;
   ~Stack();
   Stack(const Stack &original);
   void operator =(const Stack &original);
};
```

栈的实现

Stack的实现需要选择数据结构存储栈的元素。为简单起见，下面选择使用`vector<T>`。

```
class Stack {
  public:
  //  Standard Stack methods
   Stack(){};
   bool empty() const{ return elements.empty();};
   void push(const T &item) {elements.push_back(item);};
   void pop() {elements.pop_back();};
   T top() const {return elements.back();};
   //~Stack();
   //Stack(const Stack &original){};
   //void operator =(const Stack &original);
protected:
   vector<T> elements;
};
```

在这种情况下，析构函数、拷贝构造函数和重载的赋值都已经有vector提供，因此Stack不需要重新定义。但是，如果使用动态数组或者链表存储结构，则必须实现这些方法。详见[2]section 4.3。

§3 抽象数据类型的应用

标准模板库STL提供了栈的实现，它是一个模板类`stack<T>`，使用时需要提供栈元素类型参数T。

栈的特点是“后进先出”，因此，凡是遇到这种场合都要考虑使用栈。例如，判断括号匹配问题。我们先写出伪代码算法1，最后很容易将伪代码转化为代码。

算法 1 Matching(e)

输入: e是包含括号的表达式，如$e = A[i] = 2 * (x + y)$。括号包括圆括号、方括号和花括号

输出: 如果括号匹配，则输出“匹配”, 否则输出“不匹配”。

```
令S是一个空栈
for e中每个字符c do
  if c是左括号 then
    将c入栈
  end if
  if c是右括号 then
    if 如果S不空并且c与栈顶元素匹配 then
      栈顶元素出栈
    else
      return “不匹配”。
    end if
  end if
end for
if 栈S空 then
  return “匹配”。
else
  return “不匹配”。
end if
```

```
bool left(char c){
  return c=='('|| c=='[' || c=='{';
}
bool right(char c){
  return c==')' || c==']' || c == '}';
}
bool match(char a, char b){
   return (a=='(' && b==')') || (a=='[' && b==']') || (a=='{' && b=='}');
}

bool match(string cs){
  stack<char> S;//初始化空栈S;
  for (size_t i=0; i<cs.size();i++){
     if (left(cs[i]))
        S.push(cs[i]);
     else if (right(cs[i])) {
       if (!S.empty() &&  match(S.top(),cs[i]))
```

```
            S.pop();
        else
            return false;
      }
  }
  if (S.empty())
    return true;
  else
    return false;
}
```

鸣谢：感谢16级计算机专业同学们给予老师的激励，并指出文中的错漏问题!

参考文献

[1] 严蔚敏、吴伟民，数据结构，清华大学出版社，1997。

[2] Robert L. Kruse, Alexander J. Ryba. *Data Structures and Program Design in C++*, Higher Education Press, 2001.