5.1 列表的创建与删除

“呜——，火车运货来啦，第一节车厢装鸡蛋，第二节车厢装水果，第三节车厢装蔬菜，……”

列表是包含 0 个或多个数据项的有序序列，就像一列可以装载各种“货物”的列车，每节车厢放置一个数据项，如图 5-1 所示。列表的长度和内容都是可变的，可以自由对列表中的数据项进行增加、删除或替换，它没有长度限制，元素类型可以不同，使用非常灵活。

图 5-1 “火车”中的货物排列

1. 创建列表

创建一个列表，是把用逗号分隔的不同的数据项使用方括号（[]）括起来，例如：

```
list1 = ['我', '爱', '我的', '祖国']
list2 = ['eggs','fruits','vegetables','toys','clothes']
list3 = [1,2,3,4,5,6]
```

【问题 5-1】 ls = [] 是否创建了一个列表?

“[ ] 表示空列表，ls = [ ] 表示创建一个空列表 ls；
还可以直接使用 list( ) 函数来返回一个空列表。”

列表中，数据项的类型可以不同，列表的数据项还可以也是列表，例如：

```
list4 = ['我', 'fruits',101]
list5 = ['我', 'fruits',101, [1,2,'three'] ]
```

2. 删除列表的元素

“老师，列车可以卸货，列表也可删除元素吗？”

“当然可以，要删除列表元素有多种方法。”

1）使用 del 语句删除列表元素

要删除列表中指定位置的元素可以使用 del 语句，例如：

```
>>>ls=['eggs','fruits','vegetables','toys','clothes']
>>>print("原始列表:", ls)
原始列表:['eggs','fruits','vegetables','toys', 'clothes']
>>>del  ls[1]
>>>print("删除第二个元素后:", ls)
删除第二个元素后:['eggs', 'vegetables', 'toys', 'clothes']
```

2）使用 pop( ) 函数删除列表元素

pop( ) 函数可以移除列表中的一个元素（默认最后一个元素），并且返回该元素的值，例如：

```
>>>ls=['eggs','fruits','vegetables','toys','clothes']
>>>ls.pop()
```

```
'clothes'
>>>print(" 列表变为： ", ls)
列表变为：  ['eggs', 'fruits', 'vegetables', 'toys']
>>>ls.pop(0)
'eggs'
>>>print(" 列表变为： ", ls)
列表变为：  ['fruits', 'vegetables', 'toys']
```

3）使用 remove( ) 函数删除列表元素

```
>>>ls=['eggs','fruits','vegetables','toys','clothes']
>>>ls.remove("toys")
>>>print(" 列表变为： ", ls)
列表变为：  ['eggs', 'fruits', 'vegetables', 'clothes']
```

4）使用 clear( ) 函数删除列表的所有元素

例如：

```
>>>ls=['eggs','fruits','vegetables','toys','clothes']
>>>ls.clear()
>>>print(" 删除后的列表： ", ls)
删除后的列表：  [ ]
```

来找茬

以下是小萌编写的代码，功能是从列表 [5,4,3,2,1] 中删除第二个元素。

```
>>>ls=[5,4,3,2,1]
>>>ls.remove(1)
>>>print(" 删除第二个元素后： ", ls)
删除第二个元素后：  [5, 4, 3, 2]
```

“第二个元素的索引值不是 1 吗？为什么没有删除第二个元素 4 呢？”

5.2　列表的索引和访问

“老师，我知道什么是列表啦，它就像一列装载各种‘货物’的列车，但我们要怎么访问各节车厢呢？”

“与字符串一样，列表也有索引，通过索引可以方便地访问列表的各个元素。”

1. 列表的索引

与字符串的索引一样，列表索引从“0”开始，第二个索引是“1”，以此类推，如图 5-2 所示。

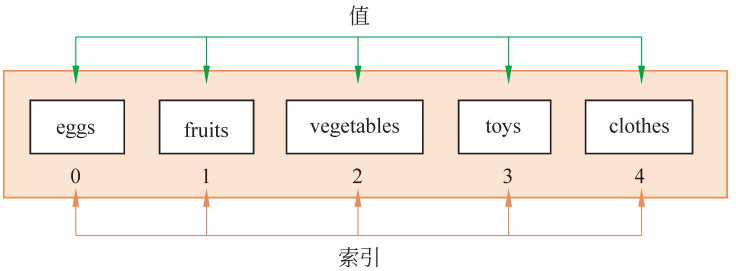

图 5-2　列表的正向索引

索引也可以从尾部开始，最后一个元素的索引为 –1，往前一位为 –2，以此类推，如图 5-3 所示。

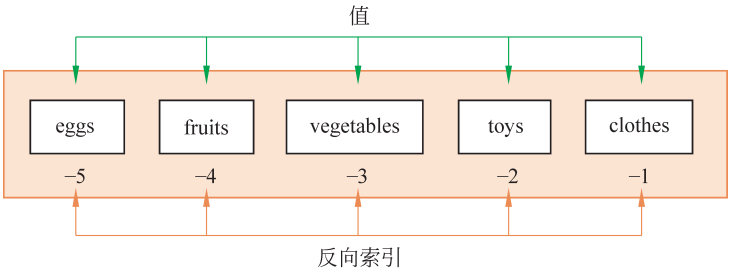

图 5-3　列表的反向索引

访问列表的值可以通过列表名加索引来实现，例如：

```
>>>ls=['eggs','fruits','vegetables','toys','clothes']
>>>print(ls[0])
eggs
>>>print(ls[1])
fruits
>>>print(ls[-1])
clothes
>>>print(ls[-5])
eggs
```

【例 5-1】 编写程序代码，实现选词填空功能，如图 5-4 所示。

继续　　连续　　陆续　　持续

观众们（ ）走进体育馆观看乒乓球赛。在男子单打比赛决定胜负的最后一分钟，两名运动员（ ）打了十几个回合才见分晓。接着，（ ）进行女子双打比赛。运动员们高超精湛的球技，博得观众一片喝彩，掌声（ ）了整整一分钟。

图 5-4　选词填空题目

程序代码如下：

```
ls=['继续','连续','陆续','持续']
print("    观众们(",ls[2],")走进体育馆观看乒乓球赛。\
在男子单打比赛决定胜负的最后一分钟，两名运动员(",ls[1],")\
打了十几个回合才见分晓。\
接着，(",ls[0],")进行女子双打比赛。\
```

```
运动员们高超精湛的球技，博得观众一片喝彩，掌声（",ls[3],"）\
了整整一分钟。"）
```

运行结果如下：

观众们（ 陆续 ）走进体育馆观看乒乓球赛。在男子单打比赛决定胜负的最后一分钟，两名运动员（ 连续 ）打了十几个回合才见分晓。接着，（ 继续 ）进行女子双打比赛。运动员们高超精湛的球技，博得观众一片喝彩，掌声（ 持续 ）了整整一分钟。

“可以通过 for-in 语句对列表元素进行遍历，基本语法结构如下：
for < 变量名 > in < 列表名 >：”

【问题 5-2】 有如下列表：

```
ls=[' 牙刷 ',' 衣架 ',' 饼干 ',' 充电器 ']
```

编写代码选出列表中不是同类的词组。

2. 列表的截取

访问列表元素和访问字符串一样，除了索引的方式，还可以使用方括号 [ ] 的形式截取。

【例 5-2】 以下是小萌数学期末考试的得分，输出她第三到第六题的得分情况：

题号	一	二	三	四	五	六	七	八	总分
得分	8	10	9	9	12	11	13	11	83

程序代码如下：

```
score=[8,10,9,9,12,11,13,11,83]
print(" 小萌数学期末考试每题的得分为：")
print(score)
print(" 小萌第三到第六题的得分分别是：",score[2:6])
```

运行结果如下：

```
小萌数学期末考试每题的得分为：
[8, 10, 9, 9, 12, 11, 13, 11, 83]
小萌第三到第六题的得分分别是： [9, 9, 12, 11]
```

5.3 列表元素的控制

列表是一个十分灵活的数据结构，它具有处理任意长度、混合类型数据的能力，并提供了丰富的基础操作符和方法。小帅和小萌玩猜拳游戏，三局两胜，要记录他们每次出拳的结果就可以使用列表来实现。

1. 增加列表元素

Python 提供了很多函数和方法来完成列表的操作，增加列表元素常使用 append( ) 函数来实现。

“小帅，我们来玩猜拳的游戏吧？”

“好啊！三局两胜。”

【例 5-3】 模拟小帅和小萌猜拳游戏的过程并记录。

分析：先创建两个空列表，准备存放小萌和小帅的出拳结果，每出拳一次在列表中增加一个元素，创建两个变量，分别存放小萌和小帅的得分，3 次出拳后得分高者胜，如图 5-5 所示。

程序代码如下：

图 5-5 猜拳游戏模拟

```
xm=[]
xs=[]
m=s=0
print("第一轮：小萌出剪刀，小帅出布")
xm.append("剪刀")
xs.append("布")
m=m+1
print("第二轮：小萌出石头，小帅出布")
xm.append("石头")
xs.append("布")
s=s+1
print("第三轮：小萌出剪刀，小帅出石头")
xm.append("剪刀")
xs.append("石头")
s=s+1
print("小萌依次出拳为：",xm,"得分为：",m)
print("小帅依次出拳为：",xs,"得分为：",s)
```

运行结果如下：

```
第一轮：小萌出剪刀，小帅出布
第二轮：小萌出石头，小帅出布
第三轮：小萌出剪刀，小帅出石头
小萌依次出拳为： ['剪刀', '石头', '剪刀'] 得分为： 1
小帅依次出拳为： ['布', '布', '石头'] 得分为： 2
```

除了 append( ) 函数外，增加列表元素还可以通过 insert( ) 函数、extend( ) 函数或“+”运算符来实现，例如：

```
>>>rainbow=["red","yellow","green"]
>>>rainbow.insert(1,"orange")
# 在列表 rainbow 索引 1 的位置加入元素 "orange"
>>>rainbow
['red', 'orange', 'yellow', 'green']
>>>t1=["indigo"]
```

```
>>>t2=["blue","violet"]
>>>rainbow.extend(t1)          # 将列表 t1 增加到列表 rainbow 中
>>>rainbow
['red', 'orange', 'yellow', 'green', 'indigo']
>>>rainbow+=t2                 # 将列表 t2 增加到列表 rainbow 中
>>>rainbow
['red', 'orange', 'yellow', 'green', 'indigo', 'blue', 'violet']
```

【问题 5-3】 将列表 s2、s3 的内容添加到 s1 中。

s1=[“富强”,“民主”,“文明”,“和谐”]

s2=[“自由”,“平等”,“公正”,“法治”]

s3=[“爱国”,“敬业”,“诚信”,“友善”]

2. 修改列表元素

修改列表元素主要有如表 5-1 所示的几种赋值方式。

表 5-1 修改列表方法

方　法	描　述
ls[i]=x	替换列表 ls 第 i 项数据为 x
ls[i:j]=lt	用列表 lt 替换列表 ls 中第 i 到第 j 项数据（不含 j 项）
ls[i:j:k]=lt	用列表 lt 替换列表 ls 中第 i 到第 j 项以 k 为步长的数据

例如：

```
>>>ls=[0,1,2,3,4]
>>>ls[0]="Sunday"
>>>ls
['Sunday', 1, 2, 3, 4]
>>>ls[1:3]=["Monday","Tuesday"]
>>>ls
['Sunday', 'Monday', 'Tuesday', 3, 4]
```