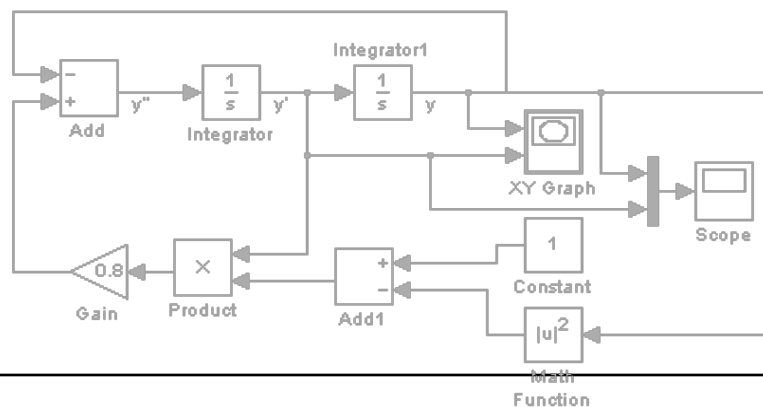


# 第九章

## 字串的處理



### 本章學習目標

認識字串

認識字串的儲存方式

學習不同數字系統之間的轉換

學習各種與字串相關的函數



# 9.1 認識字串

---

- 字串可看成是由字元所組成的陣列：

```
>> str1=['M' 'a' 't' 'l' 'a' 'b']  
str1 =  
Matlab
```

```
>> str2='I love Java'  
str2 =  
I love Java
```

```
>> size(str1)  
ans =  
1      6
```

- 
- 如要顯示字串裡每一個字元的ASCII碼，可用 `double` 函數：

```
>> ascii=double(str1)  
ascii =  
      77      97     116     108      97      98
```

```
>> char(ascii)  
ans =  
Matlab
```

## 9.2 字串陣列

---

- 用二維陣列儲存字串時，每一個字串的長度必須相等：

```
>> season=['spring';'summer';'autumn']
```

```
season =
```

```
spring
```

```
summer
```

```
Autumn
```

```
>> season(1:5)
```

```
ans =
```

```
Ssapu
```

```
>> season(1,:)
```

```
ans =
```

```
spring
```

- 
- 以二維陣列儲存字串時，如果字串的長度不相等，則會有錯誤訊息：

```
>> month=['April';'May';'June']  
??? Error using ==> vertcat  
All rows in the bracketed expression must  
have the same number of columns.
```

```
>> month=['April';'May  '; 'June  ']  
month =  
April  
May  
June
```


## 9.3 字串與數字系統的轉換

### 9.3.1 執行指令字串

- Matlab提供了eval與feval函數，可用來對字串求值：

表 9.3.1 字串求值函數

函 數	說 明
<code>eval(str)</code>	執行字串 <i>str</i>
<code>feval(func_name, arg)</code>	以 <i>arg</i> 為引數，執行函數 <i>func_name</i>
<code>feval(func_name, arg<sub>1</sub>, arg<sub>2</sub>, ...)</code>	以 <i>arg<sub>1</sub>, arg<sub>2</sub>, ...</i> 為引數，執行函數 <i>func_name</i>



---

```
>> eval('32+6')
```

```
ans =  
    38
```

```
>> eval('x=cos(pi/4)')
```

```
x =  
    0.7071
```

```
>> feval('zeros',3)
```

```
ans =  
     0     0     0  
     0     0     0  
     0     0     0
```

## 9.3.2 字串與數值的轉換

- 下表是用來進行數值與字串之間轉換的函數：

表 9.3.2 數值與字串的函數

函 數	說 明
<code>int2str(x)</code>	先將 $x$ 經四捨五入轉換成整數，再將它們轉換成字串
<code>num2str(x)</code>	將 $x$ 轉換成字串，並以 4 個位數來顯示
<code>num2str(x,n)</code>	將 $x$ 轉換成字串，但以 $n$ 個位數來顯示
<code>mat2str(x)</code>	將陣列 $x$ 轉換成 <b>Matlab</b> 的表示方式，但以字串來顯示
<code>str2num(str)</code>	將字串 $str$ 以 <code>eval</code> 函數求值，如果不能轉換，則回應空陣列
<code>str2double(str)</code>	將字串 $str$ 轉換成數值，如果不能轉換，則回應 NaN





---

```
>> str1=int2str(1024)
```

```
str1 =  
1024
```

```
>> str2num('123456')
```

```
ans =  
    123456
```

```
>> str2num('123+456')
```

```
ans =  
    579
```

## 9.3.3 不同數字系統的轉換

- 下表列出了各種進位系統之間的轉換函數：

表 9.3.3 不同數字系統的轉換函數

函 數	說 明
<code>dec2bin(<math>x</math>)</code>	將 10 進位的整數 $x$ 轉換成 2 進位的字串
<code>dec2hex(<math>x</math>)</code>	將 10 進位的整數 $x$ 轉換成 16 進位的字串
<code>bin2dec(<math>bin\_str</math>)</code>	將 2 進位的字串 $bin\_str$ 轉換成 10 進位
<code>hex2dec(<math>hex\_str</math>)</code>	將 16 進位的字串 $hex\_str$ 轉換成 10 進位
<code>dec2base(<math>x, base</math>)</code>	將 10 進位的整數 $x$ 轉換成 $base$ 進位的字串
<code>base2dec(<math>str, base</math>)</code>	將 $base$ 進位的字串 $str$ 轉換成 10 進位的整數



---

```
>> dec2bin(1122)
```

```
ans =
```

```
10001100010
```

```
>> bin2dec('10001100010')
```

```
ans =
```

```
1122
```

```
>> hex2dec('AA5F')
```

```
ans =
```

```
43615
```

## 9.4 字串處理函數

- 下表列出了常用的字串處理函數：

表 9.4.1 字串處理函數

函 數	說 明
<code>upper(str)</code>	將字串 <i>str</i> 轉換成大寫
<code>lower(str)</code>	將字串 <i>str</i> 轉換成小寫
<code>deblank(str)</code>	將字串 <i>str</i> 後面的空白字元全部刪除
<code>strcmp(str<sub>1</sub>, str<sub>2</sub>)</code>	比較字串 <i>str<sub>1</sub></i> 與 <i>str<sub>2</sub></i> 是否相等，若是，則回應 1，否則回應 0
<code>strncmp(str<sub>1</sub>, str<sub>2</sub>, n)</code>	比較字串 <i>str<sub>1</sub></i> 與 <i>str<sub>2</sub></i> 在第 <i>n</i> 個位置的字元是否相等，若是，則回應 1，否則回應 0
<code>findstr(str, s)</code>	找出字串 <i>str</i> 裡，子字串 <i>s</i> 所出現的位置
<code>strrep(str, s<sub>1</sub>, s<sub>2</sub>)</code>	將字串 <i>str</i> 裡，子字串 <i>s<sub>1</sub></i> 代換成字串 <i>s<sub>2</sub></i>
<code>strtok(str, token)</code>	將字串 <i>str</i> 裡，字元 <i>token</i> 之後的字串全都刪掉。若省略 <i>token</i> ，則以空白鍵當 <i>token</i>
<code>strvcat(str<sub>1</sub>, str<sub>2</sub>)</code>	將字串垂直排列



---

```
>> upper('Merry Christmas')
ans =
MERRY CHRISTMAS
```

```
>> str1=deblank('snoopy  ')
str1 =
Snoopy
```

```
>> findstr('kitty','t')
ans =
4
```

```
>> strvcat('hello','kitty')
ans =
hello
kitty
```



---

-The End-