

getline vs. cin input

To understand how the *getline* member function from the *apstring* class works, we first must look at how the statement

```
apstring s;  
cin >> s;
```

works. In the above code the *cin >> s* statement reads characters from the input stream until a white space is found. All characters up to but not including the white are returned in the variable *s*. The white space remains in the input stream. White space is: <space>, <tab>, <Return>.

The *getline* statement in the code below

```
apstring x;  
getline(cin, x);
```

reads in all characters up to the <Return> and returns all those characters read, including any <space> and <tab> characters. The <Return> is not returned in the string *x*, but it is removed from the input stream.

Here are some examples when you mix *getline* with *cin* statements using different data types.

Input

```
Tom is here<RETURN>  
Mary is here also<RETURN>
```

```
apstring x, y;  
int num;  
  
getline(cin, x);          // x = "Tom is here"  
cin >> y;                 // y = "Mary"  
-----  
cin >> y;                 // y = "Tom"  
getline(cin, x);         // x = " is here"  
-----  
cin >> x;                 // x = "Tom"  
cin >> y;                 // y = "is"
```

Input

```
Willie Mays<RETURN>  
24<RETURN>
```

```
getline(cin, x);          // x = "Willie Mays"  
cin >> num;               // num = 24
```

Now look what happens here!

Input

```
44<RETURN>  
Willie McCovey<RETURN>
```

```
cin >> num;               // num = 44, cin leaves the <RETURN> in the buffer  
getline(cin, x);         // x = "", getline reads everything up to <RETURN> which is nothing
```