

## Overloading Functions

Sometimes we would like a function with the same name to handle different data types, instead of using different names for the same purpose. For example, if we want to swap to data items for the int data type, and for the double data type I do not want to have a *SwapInt* function and a *SwapDouble* function, I would like to have just one function called *Swap* that handles both situations. Having one function name handling different data types is called function overloading, and in general terms is an example of polymorphism in C++.

Here is the code for three swap functions with the same name.

```
void Swap(int x, int y)
{
    int temp = x;
    x = y;
    y = temp;
}

void Swap(double x, double y)
{
    double temp = x;
    x = y;
    y = temp;
}

void Swap(apstring x, apstring y)
{
    apstring temp = x;
    x = y;
    y = temp;
}
```

From a program we can now do

```
int main ()
{
    int a = 4, b = 7;
    double r = 4.4, s = 5.5;
    apstring s1 = "Rowan", s2 = "Martin";

    Swap(a, b);
    Swap(r, s);
    Swap(s1, s2);
}
```

The compiler determines which of the three *Swap* functions to call by the number of parameters, the type of the parameters, and the order of the parameters. Overloaded functions are determined by their *signature*, the combination of the number of parameters, their types, and order.