When to Use Function Overloading

You might find function overloading fascinating, but don't overuse the facility. You should reserve function overloading for functions that perform basically the same task but with different forms of data. Also, you might want to check whether you can accomplish the same end with default arguments. For example, you could replace the single, string-oriented `left()` function with two overloaded functions:

```c
char * left(const char * str, unsigned n); // two arguments
char * left(const char * str);           // one argument
```

But using the single function with a default argument is simpler. There's just one function to write, instead of two, and the program requires memory for just one function, instead of two. If you decide to modify the function, there's only one you have to edit. However, if you require different types of arguments, default arguments are of no avail, so then you should use function overloading.