

# CS113 - Program Design and Algorithms

## Lab Assignment #4

### Summer 2001

The purpose of this exercise is to give you a chance to write a C++ class. The class you will write for this exercise can be based on the work you did for program #3.

For this exercise you are to create a class with methods that accept "numbers" in the form of strings, stores them as numeric values within the class, and outputs them in the various bases. The numeric value should be a private member of the class; all public references to this value should use *methods* that use string forms. These methods should do the somewhat complex task of converting the strings to numeric, and numeric back to string.

Therefore, at a minimum, for this exercise you should define a class with the following methods:

`int Numinput(float val);` - should define the value of the class with a numeric value. This method can be used to define the class value with a number, and may be useful before the string input methods (below) are written. The value returned should be "true" if the conversion succeeded, and "false" if some error occurred.

`int Bininput(char str[]);` -  
`int Octinput(char str[]);` -  
`int Decinput(char str[]);` -  
`int Hexinput(char str[]);` - should accept a *string* containing the characters that make up a "number" in binary, octal, decimal, and hexadecimal, respectively, and define the value with the numeric value represented by the string. Return value should be "true" if the conversion succeeded, "false" if not.

`void Binoutput(char str[], int prec);` -  
`void Octoutput(char str[], int prec);` -  
`void Decoutput(char str[], int prec);` -  
`void Hexoutput(char str[], int prec);` - should output the characters that represent the class value, in binary, octal, decimal, or hexadecimal, respectively, to standard output. The value of `prec` determines the number of places behind the radix point to output. The value of zero should output the radix point but **NO** places behind the point; a negative value should output only the integer part of the number, without any radix point.

Additional information:

- Before a value is defined, it should be initialized to a value of 0.
- If some error in defining the value occurs, then the value should remain unchanged.
- The actual value itself should be hidden from the user (i.e., main program), accessed only through the above methods.

- The means by which the actual numeric value is stored within the class is up to you. Some ways of storing the value are probably going to be easier to deal with than others!

Show that your new class and its methods work by showing the results from the following tests:

1. Input a value in each of the four bases as a string, and output it in all four bases.
2. Input some “illegal” values in each of the bases, and have your program output an error message.

Assignment