

CSE 320 Spring 2012

Computer Project #5 -- Number Systems and Integer Representation

Assignment Overview:

This assignment develops familiarity with the C programming language, the "gcc" compiler, number systems, and twos complement representation.

It is worth 40 points (4% of course grade), and must be completed no later than 11:59 PM on Thursday, March 1.

Assignment Specifications:

1) You will develop a C function which supports the conversion of signed integer values between internal representation and external representation. The prototype for the function is:

```
int convert( int, unsigned, char[], int );
```

The first argument is the value which is to be converted into external representation.

The second argument is the base into which the value is to be converted. Valid bases range from 2 to 36 (inclusive).

The third argument is the address of an array where "convert" will store the external representation of the value as a null-terminated sequence of ASCII characters. That representation will be in the form "N base B", where "N" is the sign ('+' or '-') and the sequence of digits representing the value, and "B" is the base.

The fourth argument controls the appearance of the external representation. The absolute value of the argument specifies the total number of characters in the "N" field (including leading zeroes or blanks). If the argument is negative, the conversion will use leading zeroes to fill the "N" field (if needed), and the sign character will be the leftmost character in the "N" field. Otherwise, the conversion will use leading blanks to fill the "N" field, and those blanks will be the leftmost characters in the "N" field.

The function will return the value one if the conversion is successful, and the value zero otherwise.

2) You will develop a driver module to test your implementation of the conversion module. The source code for the driver module must be in a separate file. All output will be appropriately labeled.

Your driver module will not be interactive. If it accepts any input, you will supply that input in a text file named "proj05.tests".

Assignment Deliverables:

The deliverables for this assignment are:

```
proj05.makefile -- the makefile which produces "proj05"  
proj05.support.c -- the source code for your conversion module  
proj05.driver.c  -- the source code for your driver module  
proj05.tests     -- the input to your driver module, if needed
```

Be sure to submit your files for grading via the "handin" program.

Assignment Notes:

- 1) Chapter 2 of the Murdocca and Heuring text contains relevant information about number systems (particularly pages 22-24).
- 2) Please note that your source code must be translated by "gcc", which is a C compiler and accepts C source statements (a subset of C++).
- 3) You may not call any C library functions from your conversion module. That is, you may not call functions such as "scanf", "atoi", or "isspace" from function "convert" or any other functions that you choose to write as part of your conversion module.

Your conversion module must convert between integers and characters without using any C library functions. The equivalent integer value of a character from the set {0..9} representing a decimal digit may be obtained by:

```
char ch;
int value = ch - '0';
```

Similarly, the equivalent integer value of a character from the set {A..F} representing a hexadecimal digit may be obtained by:

```
char ch;
int value = (ch - 'A') + 10;
```

- 4) Please note that your driver module may not be written as an interactive program, where the user supplies input in response to prompts. Instead, your test cases will either be included in the source code as literal constants, or will be supplied as inputs to the driver module in a file named "proj05.tests".

If your driver module requires no inputs, your solution will be executed using:

```
proj05
```

If your driver module does require inputs, your solution will be executed using:

```
proj05 < proj05.tests
```

- 5) Consider the following, which shows the character strings produced by a series of successful conversions.

```
int A = 127, B = -6;
char C[80];

convert( A, 10, &C[0], 5 ); ==> C contains " +127 base 10"
convert( A, 10, &C[0], -5 ); ==> C contains "+0127 base 10"
convert( A, 16, &C[0], 5 ); ==> C contains " +7f base 16"
convert( A, 16, &C[0], -5 ); ==> C contains "+007f base 16"
convert( B, 7, &C[0], 3 ); ==> C contains " -6 base 7"
convert( B, 7, &C[0], -3 ); ==> C contains "-06 base 7"
```

For unsuccessful conversions, the character string should be empty.