CST 334: Operating Systems

Dr. Glenn Bruns

# Binary to decimal conversion

**Purpose**: Operating systems often work with bits and bytes at a low level. It's important for you to become comfortable with binary and hexadecimal representation of numbers. The purpose of this assignment is to help you get more familiar with translating numbers between bases, and at the same time to build your C programming skills.

**Instructions**: You will write a program bindec.c that will take a binary numbers as input, and output the decimal equivalent. I provide code you will use as a starting point.

1. On mlc104, copy bindec.c to a directory of your own. For example:

$ cp /home/CLASSES/brunsglenn/cst334/hw/hw1/bindec.c <yourdir>

Note: if you don't yet have access to mlc104, the files you need for this assignment can be found [here](https://drive.google.com/file/d/181_UHmPeI59qZENmxUL_5I0vgIalLlk-/view?usp=sharing). You can work on this assignment on any Linux system.

1. Use a text editor (like vim or nano) to edit the code. Look for "YOUR CODE HERE". Modify bindec.c so that it converts the input string into a number, and prints the number. Write the code to convert from binary to decimal yourself (don’t use a library call, or code found on the internet).
2. Your code should respond to empty input with value 0, and should respond to invalid input with message ‘input must contain only zeros and ones’, and exit with value 1, indicating an error. Your code should be able to handle inputs of at least 20 binary digits.

Some examples of how your program should work:

$ ./bindec

> 1001

9

$ ./bindec

> 11111111

255

$ ./bindec

> hello

input must contain only zeros and ones

$ ./bindec

> (no input given, just hit ‘enter’)

0

Testing your code**.** In the directory /home/CLASSES/brunsglenn/cst334/hw/hw1 there are some scripts you can use to test your code. The scripts assume your code is named 'bindec.c'. Here's how to run a test:

$ ./test1.sh

$ echo $?

The result of the second command will be 0 if your code is correct, and some other number if your code is not correct. You need to run test1.sh first; it will test to see if your code compiles, and will generate a binary named bindec that the other test scripts will use.

If you are not running the tests on mlc104, make sure that test1.sh and the other test scripts have "execute permission". You can run this command to give them execute permission:

$ chmod +x test\*.sh

The tests I supply are sample tests. In grading I may use different tests and additional tests.

Tidy code: Your code should be indented properly and consistently, should use blank lines and other whitespace consistently, and should contain the appropriate amount of comments: at least one comment at the top of each file, and one comment just before each function. See the [tidy code slides](https://drive.google.com/file/d/13d3-TsC7NVIPRLVo1EYHpKFzFyKwd1qx/view?usp=sharing) for more information.

**Submission**. Submit your file bindec.c on iLearn. If you need to copy code from or to mlc104, please refer to these [instructions](https://docs.google.com/document/d/1TY6waCCHFkSBYCWKsHUelg9KH2yjXS2VOAyZLIPu93U/edit?usp=sharing).

**Grading**. 10 points for compilation, 5 points for each of 4 additional tests, and 10 points for tidy and properly formatted code. No points for compilation or tidy code if your code is obviously incorrect.