

Part I – Programming

You need to create a program that will take a number from standard in, in the form of a string and convert that string to its binary representation. The number entered will be in decimal format.

For example, the user enters a number and it is read in as a sequence of characters. If the number “5” is entered then you will convert the string to the binary number “101”.

After requesting the first number from the user, you will need to convert it and display that number back to the screen by converting the binary number to a string representation. The string shall display the binary number in the form of 1’s and 0’s. After displaying the converted number you will ask the user for another number that will be the exponent used in raising the first number to. With the first number still in binary formation you will raise it to that exponent and display the result back to the screen. You will have to take the requested exponent a likewise fashion as the first number. That is, you will take the number from standard in as a string and convert the string to its binary representation in order to use it.

You will HAVE to create at least three functions to perform these requirements: `converttobin`, `converttoascii`, and `pow`. `Converttobin` will take a string as an input in and return a number that is the binary conversion of that string as an output. `Converttoascii` will take a number as an input and return a string representation of that number as an output. The `pow` function will take a number and its exponent as an input and will return a value representing that number raised to that exponent as an output. These inputs MUST be passed as parameters. The `pow` function MUST be recursively implemented (NOTE: this is not the most efficient way).

You are NOT allowed to use “PutDec,” or “GetDec” in any part of this function for the purposes of obtaining an integer from input. You are also NOT allowed to use any predefined macros or procedures for displaying information to the screen. You may ONLY USE the interrupts for displaying to the screen. In other words, you are NOT allowed to use anything but your own ingenuity.

Your program will repeat a certain number of times as specified by the constant “REPEATTIMES” that can be found in an include file “constants.inc” You will need to included this file using the “include” directive in your program. You can expect that the incoming numbers will be non-negative. That means you will have to do error checking to verify that a correct number has been given. If an incorrect number has been given, (i.e. “-5”) then you prompt for that information again, and only that information which is incorrect.

You are also required to prevent overrun out of any register. That is if the result of the power function is greater than a single 16 bit register can handle, then you return from the `pow` function with the last number raised. If you use the 32 bit set, you will have to return the last number raised before a that register is overrun. In other words, if you use either the 16 bit or 32 bit registers, you are still required to perform the same tasks.

