Data Representation and Computer Arithmetic

Programming Assignment 2
CS 211: Computer Architecture, Fall 2016

By: Michael Russo
Net Id: mr880
Due: 10/21/16

calc.c

To start, I created structures to store arrays and variables for inputs and performs an initial check using **int initialCheck(int arg)**.

Taking those inputs, I used a function **int funnel(char* input)** to 'funnel' input numbers into their proper conversion functions while checking for missing inputs as an Error check and for whether or not the value entered is a negative input.

int bin2dec(int len, char* input)
int dec2dec(int len, char* input)
int oct2dec(int len, char* input)
int hex2dec(int len, char* input)

These are the functions the inputs were sent to convert our values into decimal integers The use of Switch statements were especially necessary in order to have clean systems of conversion from type char to int.

The values returned then go into the **int math(int A, int B, char op)** function in which both input values, now in integer decimal form, along with the desired operation type to be performed, are sent into a simple algorithm in which the math is performed and returned an int.

Using an if statement, based on what char was passed through for the output function, these four functions were called…

void int2bin(int output)
void int2dec(int output)
void int2oct(int output)
void int2hex(int output)

… in which an array struct was used in each case to store our values int a 32 bit sequence and performed further conversions…

void dec2binConv(int input, int* arr)
void dec2OctConv(int input, int* arr)
void dec2HexConv(int input, int* arr)

void dec2DecConv(int input, int* arr)

… in order to put their values into arrays.

void joinArrsDec(int *arr2, int *arr1)
void joinArrs(int *arr2, int *arr1) (binary)
void joinArrsOct(int *arr2, int *arr1)
void joinArrsHex(int *arr2, int *arr1)

These functions were called next to join the arrays in our struct.
Still in our int2bin etc… functions, I removed remainders, leading zeros and printed values to the user for an output, ending the program save the garbage collection.

For a big-O analysis, nested while loops provide an O(n) performance analysis.