calc

Special Features for extra credit:

1. I've implemented multiplication. It works for any base. The product must fit in 32 bits. My logic for multiplication is: Take two numbers, convert to decimal, multiply in decimal, convert to output base.
   Note: for multiplication, my code works only if you put double quotes around the * sign, otherwise there will be an error message saying "Invalid input." This is because the terminal reads * as a special value (like ^ or &).
2. My code handles arbitrarily large values for adding numbers in the same base because I handle the numbers in char* without needing to convert base.
3. I increased efficiency by calling conversion code only if necessary.

General logic for add and subtract: Take two numbers, convert to binary, perform operation, convert to output base.

Big O and Space analysis for the functions:
checkValidNumber = O(n) where n is the length num
initCalculate = O(1). Space used is the size of struct Calculate
append = O(1) also time to call realloc
StringToInt = O(n) where n is the length of word
DeciToBin = O(n) where n is the length of num
getBinNumber = O(1)
HexToBin = O(n) where n is the length of num
IntToString = O(n) where n is the length of num
BinToDeci = O(n) where n is the length of num
BinToOctal = O(n) where n is the length of num
getHexLetter = O(1)
BinToHex = O(n) where n is the length of num
ConvertBinaryToOutput = O(1)
insertZeros = O(n) where n is the length of num
add = O(n) where n is the length of the longer number
subtract = O(n) where n is the length of the longer number
baseToDecimal = O(n) where n is the length of num
AdditionControl = O(1)
SubtractionControl = O(1)
multiply = O(1) only function calls
MultiplicationControl = O(1)
performFunction = O(1)
Destroy = O(1)
main = O(1)

Test cases:
Input: + -b10001110101 o123 x
Output: −x422
Input: + d1111111111111111 d1111111111111111 d

Output: d2222222222222222
Input: # d1 d1 d
Output: ERROR: Invalid operation.
Input: “*” x5Aef o375 o
Output: o26357063
Input: “*” d46340 d46340 d
Output: d2147395600

float

Big O and Space analysis for the functions:
initFormat = O(1). Space used is the size of struct Format
Integer_BinToDecimal = O(n) where n is the length of num
append = O(1) also time to call realloc
getMantissa = O(n) where n is the length of num
checkINForNAN = O(1) also time to call strcmp
Float_BinToDecimal = O(n) where n is the length of number to convert to decimal
convertToDecimal = O(1) also time to strcmp
Destroy = O(1)
main = O(1)

Test cases:
Input: 10000001010000100100001101000100 int
Output: -2126363836
Input: 11111111101010101010011101010100 float
Output: -NaN
Input: 12874911111111111111000000024890 float
Output: ERROR: Invalid input bit sequence.