

CS111 - Introduction to C Programming

Fall 2025

Programming Assignment #8 (3%)

Due: Friday November 14 (23:59)

C Recursive Functions

Objectives

In this programming assignment, you will continue to learn how to create C functions including recursive functions.

Part I: Activation functions of a neural network in C (1%)

Neural networks are a simple yet powerful computational model that is fundamental to the development of modern-day artificial intelligence. The basic component of a neural network is an artificial neuron that accepts as input a number of values whose weighted sum is processed by an activation function to determine the output of the neuron. Given a weighted sum, x , four common activation functions are: sigmoid $\sigma(x)$, hyperbolic tangent $\tanh(x)$, rectified linear unit $\text{ReLU}(x)$ and leaky rectified linear unit $\text{Leaky ReLU}(x)$, defined respectively by:

$$(1) \quad \sigma(x) = \frac{1}{1+e^{-x}} \quad (2) \quad \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}},$$
$$(3) \quad \text{ReLU}(x) = \max(0, x) \quad (4) \quad \text{Leaky ReLU}(x) = \max(0.1x, x)$$

Create four C functions that implement these four activation functions. The four functions are to be called from a main program shown below to generate their activation values. Exponential function `exp(x)` is declared in `math.h` and `max(x,y)` should return the larger of `x` and `y`.

```
# include <stdio.h>
# include <math.h>

float sigma(float), my_tanh(float), relu(float), l_relu(float);

int main() {
    float x;

    scanf("%f", &x);
    printf("%f %f %f %f\n", sigma(x), my_tanh(x), relu(x), l_relu(x));
    return 0;
}
```

Part II: Printing of a positive decimal number (2%)

In this part of the assignment, you will write a function to print out the digits of an integer recursively. Specifically, given an n-digit decimal number, the solution can be expressed as printing out the leading n-1 digits first and then the least significant digit. This is where a recursive solution can be applied, as printing an n-digit number (original problem) is solved by printing an (n-1)-digit number (simpler problem), followed by printing out the the least significant digit (the base case). Whether a number is a single digit can be determined by an

integer division of the number by 10. *You program needs to handle both positive and negative numbers.*

Use the following main program as a start to test your solution. Then modify it if necessary in order to handle negative numbers.

```
# include <stdio.h>

void print_digits(int);

int main() {
    int i;

    printf("input a number: ");
    scanf("%d", &i);
    print_digits(i);
    printf("\n%d\n", i);
    return 0;
}
```

The expected behavior of your program is provided below:

```
% ./p8p2
input a number: 1784
1784
1784
% ./p8p2
input a number: -34899
-34899
-34899
```

Marking and Assignment Submission

Once the program is completed, add a header documentation section and inline comments to explain how the program works. The header section should include information about the author and the general information about this assignment as well as about the use of AI as described in our “AI policy”. Each function should have its own header as the main function to describe the input and output the function and the algorithm used to compute output from the input. (This is called “proper documentation” below.) Name the programs for the two parts as **pa8p1** and **p8p2.c** respectively. Submit **pa8p1.c** and **p8p2.c** as well as the zipped folder of the three files, called **pa8.zip**, as three separate files through Blackboard. Be prepared to explain to a TA in the next lab session about your solution to this assignment. This programming assignment is worth a total of 3% with 1% for Part I and 2% for Part II.

Please work on the homework independently. The university has a zero-tolerance policy on plagiarism. Regarding the use of AI to assist you in completing assignments, please refer to “AI Policy” on course Blackboard.