System Skills and Low-Level Programming Review

Nawat Ngerncham

November 13, 2022

Contents

1 C Programming Language

1.1 Pointers

Now that we know how to program in C, let us take a look at one of the most painful concepts used in C programming – pointers.

Pointers are pretty self-explanatory, really. They point to an address in memory. The annoying part comes from the fact that this address could be invalid, meaning that it has not been allocated or is not owned by you. This then gives rise to the well-known segmentation fault errors that we all *love*.

Before we learn about how to use pointers, let us first look at some concepts related to pointers.

1.1.1 Arrays

An array is a collection of data of the same data type and is one of the more simple data structures out there. Interestingly, strings are also stored as arrays in C (arrays of `chars`, specifically). If we could somehow zoom into the memory, we will see that the memory allocated for an array is contiguous and is in the data area of DRAM.

Arrays in C

```c
int i[10]; // array of ints of length 10
char str[20]; // array of chars or string of length 20
```

Interestingly, since C is such a low-level language, strings in C are actually declared as an array of `chars` that terminates with the null character or `'\0'`. For example, the string for `"Hello World!"` could be visualized as:

```c
char hello[13] = {'H', 'e', 'l', 'l', 'o', ' ', 'W', 'o', 'r', 'l', 'd', '!', '\0'};
printf("%s\n", hello); // Hello World!
```

1.1.2 Structs

For anyone familiar with object-oriented programming, structs can be thought of as classes that only contains instance variables, no methods. For those who are not familiar with OOP, think of structs as a box where you can put multiple pieces of data in it.

Structs in C

To declare a struct in C, simply use the keyword `struct` followed by its name and the data you would like to put inside it.

```c
struct foo_struct {
    int a;
    float b;
};
```

To access the members of a struct, simply use dot notations like how one would access the instance variables of an object in Java.

```
foo_struct.a; // gets the value of a
foo_struct.b; // gets the value of b
```