Systems Programming

A (Soft) Introduction to C, Part II

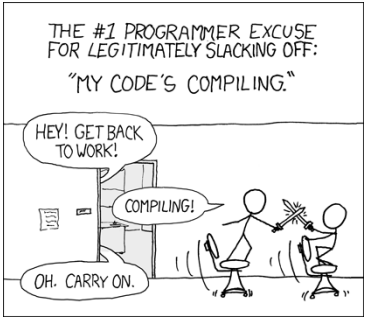
Vassilis Markos, Mediterranean College

Week 06

Contents

C Functions

Compiling Stuff...



Breaking news from Developers' Land. Source: `https://xkcd.com/303/`.

Writing Some Code

What does this C++ code compute?

```c
// source/someCode.c
#include <stdio.h>

int main() {
    int n;
    printf("Please, enter a positive integer: ");
    scanf("%d", &n);
    int i = n;
    int f = 1;
    while (i > 0) {
        f *= i;
        i--;
    }
  printf("%d! == %d\n", n, f);
}
```

How Can We Repeat This Twice?

```c
// source/someNaiveCode.c
#include <stdio.h>

int main() {
    int n;
    printf("Please, enter a positive integer: ");
    scanf("%d", &n);
    int i = n;
    int f = 1;
    while (i > 0) {
        f *= i;
        i--;
    }
    printf("%d! == %d\n", n, f);
    printf("Please, enter a positive integer: ");
    scanf("%d", &n);
    i = n;
    f = 1;
    while (i > 0) {
        f *= i;
        i--;
    }
    printf("%d! == %d\n", n, f);
}
```

What About Loops?

```c
// source/whatAboutLoops.c
#include <stdio.h>

int main() {
    int n;
    int i;
    int f;
    for (int j = 0; j < 5; j++) {
        printf("Please, enter a positive integer: ");
        scanf("%d", &n);
        i = n;
        f = 1;
        while (i > 0) {
            f *= i;
            i--;
        }
    printf("%d! == %d\n", n, f);
    }
}
```

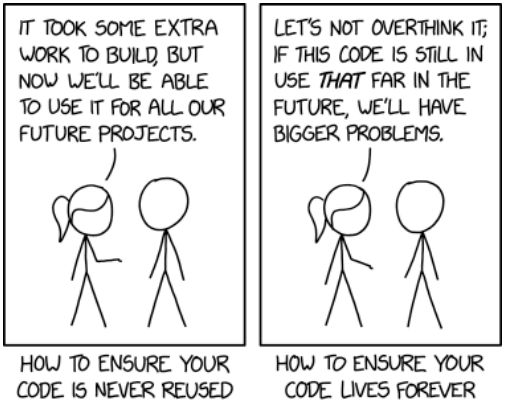
Abstraction Through Functions

```c
// source/functionAbstraction.c
#include <stdio.h>

int factorial(int n) {
    int i = n;
    int f = 1;
    while (i > 0) {
        f *= i;
        i--;
    }
    return f;
}

int main() {
    int n;
    int f;
    for (int j = 0; j < 5; j++) {
        printf("Please, enter a positive integer: ");
        scanf("%d", &n);
        f = factorial(n);
    printf("%d! == %d\n", n, f);
    }
}
```

Why Use Functions?

- **Code Maintenance:** It is way easier to maintain your code if all functionality that is intended to be reused is defined and kept at one place.
- **Code Distribution:** Imagine reading a cryptic project where everything is defined at difficult to spot places. Avoid this for your projects as much as possible.
- **Debugging:** The more concise your code the easiest to understand and, consequently, the easiest to debug.

A Brief Reality Check



Another day in office. Source: `https://xkcd.com/2730/`

Function Terms

- **Function signature:** The part of the function definition that includes:
  - Its name, which may be any alhpanumeric string starting with a letter.
  - Its return type, which is any C valid type.
  - Its arguments, which is a list of typed variable names.
- **Return value:** Every function can return up to one variable, whose type should match the function's return type.

What's Wrong Here?

Execute the following program. What goes wrong?

```c
// source/problem_001.c
#include <stdio.h>

double foo(int x) {
    double y = x / 2;
    return y;
}

int main() {
    printf("%f\n", foo("5"));
}
```

What's Wrong Here?

Execute the following program. What goes wrong?

```c
// source/problem_002.c
#include <stdio.h>

double foo(int x) {
    return "f";
}

int main() {
  printf("%f\n", foo(5));
}
```

What's Not Wrong Here?

Execute the following program. Why does it work?

```c
// source/problem_003.c
#include <stdio.h>

double foo(int x) {
    return 'f';
}

int main() {
  printf("%f\n", foo(5));
}
```

What's Wrong Here?

Execute the following program. What goes wrong?

```c
// source/problem_004.c
#include <stdio.h>

void foo(int x) {
    return x + 4;
}

int main() {
  print("%d\n", foo(12));
}
```

What's Not Wrong Here?

Execute the following program. Why does it work?

```c
// source/problem_005.c
#include <stdio.h>

double foo(int x) {
    double y = x / 2;
    return y;
}

int main() {
  printf("%f\n", foo(6.8));
}
```

Important Notes On Functions

- Functions that do not return a value should be declared as `void`.
- C will try to cast types whenever it can. For instance, C can cast doubles to integers, so it will, if prompted to.
  - We will discuss about all of that stuff in more detail throughout this course.
  - If you feel an urge to learn more, you can look up `rvalue` and `lvalue`.
- Always remember that characters in C can also be treated as integers, with the corresponding pros and cons.

Will This Work?

Will this work? Answer before executing the program first!

```c
// source/foo.c
#include <stdio.h>

int foo(int x) {
    return x + 5;
}

void foo(char x) {
    printf("%c\n", x);
}

int foo(double x) {
    return x + 5.06;
}

int main() {
    printf("%d\n", foo(6));
    foo('3');
    printf("%d\n", foo(5.8));
}
```

Function Overloading

- Many languages allow us to define functions so long as they have a different signature.
- This means that two functions sharing the same name should either accept different types / number of arguments and / or have a different return type.
- **C does not natively support function overloading!**
- There are three ways to bypass this:
  - Use different names for each overloaded version (...).
  - Use the C11 (or newer) `_Generic` compile–time operator (https://port70.net/ nsz/c/c11/n1570.html#6.5.1).
  - Use `void` pointers and run all sort of risks (what we commonly do).

Fun Time!

In–class Exercise #001

Write a C function that:

- takes a single integer as an argument, and;
- returns `1` or `0` depending on whether this number is even or odd.

Demonstrate the functionality of your function by properly using it in a simple C script.

In–class Exercise #002

The Fibonacci numbers, f_n, are a sequence of integer numbers given by the following relation:

$$f_n = f_{n-1} + f_{n-2}, \quad f_0 = 0,\ f_1 = 1.$$

That is, each term is the sum of its previous two. For instance, the first 10 Fibonacci numbers are:

$$0, 1, 1, 2, 3, 5, 8, 13, 21, 34.$$

Write a C function that takes n as input and prints the n–th Fioinacci number, f_n.

In–class Exercise #003

Write a C program that:

- asks the user for consecutive positive integers (non–positive input terminates number insertion), and;
- computes and prints their sum and average.

You are required to use **at least three different functions** for your solution and explain your rationale!

In–class Exercise #004

The standard deviation, s, of a set of n numbers $x_1, x_2, \ldots, x_n$ is computed by the following formula:

$$s = \sqrt{\frac{(x_1 - \mu)^2 + (x_2 - \mu)^2 + \cdots + (x_n - \mu)^2}{n}},$$

where μ is the mean value of those numbers.
Write a C program that asks the user for some non–zero numbers (insertion terminated by inserting 0) and computes their standard deviation. Make sure your program uses **at least two functions!**

In–class Exercise #005

A string is said to be a **palindrome** if it reads the same left–to–right and right–to–left. Write a C function that:

- takes a single string as an argument, and;
- returns `True` or `False` depending on whether this string is a palindrome.

Demonstrate the functionality of your function by properly using it in a simple C script.

In–class Exercise #006

One way to estimate the square root of a positive float, a, is to use the following method:

$$x_n = \frac{1}{2}\left(x_{n-1} + \frac{a}{x_{n-1}}\right),$$

where the first estimate, x_0, is an arbitrary positive float. We say that x_n is an estimation of $\sqrt{a}$ of accuracy $\varepsilon > 0$ if $|x_n - x_{n-1}| < \varepsilon$, i.e., if the two latest estimates we have made are no further apart than ε.
Write a C function that takes x_0, a, and ε as arguments and returns the corresponding estimate, x_n.

In–class Exercise #007

The *Towers of Hanoi* is a well–known puzzle where you have to move disks of different sizes one at a time from a peg to another peg with the help of an auxiliary peg and without ever moving a larger disk on top of a smaller one. You can familiarise yourselves with the game below:

`https://www.mathsisfun.com/games/towerofhanoi.html`

Develop a C function that accepts a positive integer n corresponding to the number of disks on the first peg and prints on screen the required steps to solve the problem.

Any Questions?

Do not forget to fill in the questionnaire shown right!

https://forms.gle/dKSrmE1VRVWqxBGZA