A Guide to Intuitive Programming

By Parth Iyer

Contents:

Part 0: Binary Logic

All of us are familiar with the basic Arithmetic operations addition subtraction multiplication and division. However, in the world of computers, numbers are not on a scale from 1-10, they exist as 1 & 0 or True & False. This means that we can do 1 of 2 things: 1)Chain together binary digits(bits) to make a number and 2)Use our bits for logic.

The 1st is fairly simple. Take a number in base 10: 542. If we break this number down what do we see?

2 - ones
4 - tens
5 - hundreds

But if we look closer, we can see

2 - ones
4 - 10

5 - 10^2
because 100 is really just 10^2. Similarly, we can write binary numbers using bits. let's take 1011 and break it down

1 - ones
1 - twos
0 - fours
1 - eights

Seeing this might be a bit confusing at first, but in reality, it makes sense. We always start with the ones. Then we go to the base so in our case 2 (bi means 2). Then we go to 2^2 and then to 2^3 so on ad infinitum (for infinity).

Part 1: Computer Hardware

This section may seem completely unnecessary to this programming course, however, it can never hurt to know a bit about the platform upon which all your code will be built.

The computer is at the core of computer science (hence the name). So what exactly is a computer? We can think of it as a magical box on which we can run our code, use the internet, and play games. But the computer is a bit more than a magic box. It’s one we can take apart and understand.

The center of the computer is the Central Processing Unit or CPU (hence the name). It is attached to the motherboard or main board which is a series of interconnects between various different components. Then, there is the Graphical Processing Unit or GPU that draws out the screen all at once

so you can see it. Other components such as memory (RAM) and Storage help to store data for the CPU & GPU while they work on parts of a task.

Many modern programming languages automatically help you utilize the hardware appropriately, but there are cases where it is faster to use the GPU on a calculation, or the CPU on a drawing. For the purpose of learning computer science, this concept is not too important but it is good to keep in the back of your head.

Part 2: Data Types

Data is everything. All information that we receive is data. Everything from colors to textures is covered by the magical idea called a data type. A data type is a fancy word for a certain type of information.

We have all asked the question, "What time is it?" This is stored in a value called 'datetime'. Your computer uses this to store the date and the time.

Examples of 'datetime':

08/10/14,09:12:53

There are other data types as well! We have numbers, both integers (int) and decimals (floats). Examples of ints are 1 and -5, and examples of floats are -3.2 and 5.4.

Dictionary of Basic Datatypes:

datatype - a type of information

int (integer) - an integer

float (decimal) - a decimal number

char (character) - a single letter, symbol or character. Refer to Unicode Character Table for a list of these symbols

str (string) - a word, sentence, paragraph, or any sequence of characters.

bool (boolean) - true or false, named after George Boole.

datetime - date and time value

img (image) - a picture

array - a list of any other datatypes

Note: Data types help differentiate information. If your typing was “657689”

then it would be difficult for the computer to understand if you meant 657,689 or if you meant the numbers 6, 5, 7, 6, 8, and 9 individually.

Part 3: Variables

Variables sound scary. But, they are just numbers in letters, like in math. Think about a filing cabinet. Whenever a variable is set, pull out a file, label it, and put a number inside. Now, whenever the computer needs to, it can look into the folder with the variable and retrieve the value. The filing cabinet is an analogy for the computer's memory.

How can we put a number inside of a word? Here's an example: int a = 5; Now, every time we type "a" on its own, the computer knows to replace it with "5". The following example demonstrates this:

```
int number = 5
number + 45 = 5 + 45 = 50
```

Just like these numbers, we can also fit all other datatypes inside of variables.

Examples:

str a = “Hello”

What is a + a + “A”?
“HelloHelloA”

Note: Variables are case sensitive!
Challenge: Use a datatype to store 1.4 in a variable.

Part 4: Simple Commands

Finally, the moment you have been waiting for! In this section, you will learn how a program actually does something. In human language, there are commands like “Sit down” and “Eat.” Just like in English commands tell humans what to do, commands in programming tell the computer to do something. For example, a *print* command will tell the computer to display text on the screen.

Pseudo Code Documentation (Basic):

What is pseudocode? Semantically, pseudo means fake, and code means code. Pseudocode, then, is fake code. Indeed, pseudocode is code but written in English. Still, there are some rules for pseudocode. You need to use only the basic commands to

do more complex tasks. Those will be covered in this section!

Documentation:

`print(datatype)` - prints out any datatype or variable specified in parentheses

`input()` - takes and input from the user

`length(datatype)` - finds length of datatype or variable specified in parentheses

`str(datatype)` - converts a datatype to a string

`int(datatype)` - converts a datatype to an integer

`float(datatype)` - converts a datatype to a float

fill(red, green, blue) - Fills shape commands called after it with the RGB color specified.

rect(x,y,length,width) - draws a rectangle top left (x,y) and draws out side lengths l and w

ellipse(x,y,xRadius, yRadius) - draws out an ellipse with center (x,y) and radii xRadius and yRadius.

line(x_1,y_1,x_2,y_2) - draws out a line between both points on the Cartesian Plane

Part 5: Math in Programming

Math is an essential part of programming. We see it a lot when printing and drawing. With programming, we can create calculators that use variables and operations. Math has many operations, and in programming we sometimes use these operations.

Some operations do change in math, most notably squares. Normally, to take a number to a certain power, it is written as a^b. In programming, it is written as a**b. In addition, most languages have a Math function (spelled as in the text) which contains most all math functions and usually is included in the languages Documentation. With it, a**b also equals Math.pow(a, b).

Part 6: The Conditional

Decisions are at the core of programming. At the core of these decisions, the computer checks if a boolean expression is true. If it is, the computer runs code that it wouldn't have run otherwise. We see this in an "if" statement with the following formatting:

```
if (conditional) {
    Code to run if conditional is true
}
```

This is the crudest example of an if statement, and to accompany this we have and else statement.

```
if (...) {
    ...
}
else {
    output
}
```

There are also else if statements or elifs.

Example:

```
if (a == b) {
    a = a + 1;
}elif(a+b == a^b){
    a = b^(a^b);
}else{ // This is ok code. Also, declare
         // comments with "//". Comments
       are
         // ignored by the computer.
    a = b
}
/*
This
is a multi line
comment
*/
// use comments appropriately.
```

Assignment:
Check to see if a number is equal to a number. Take inputs using the following Code:

```
var = input(String)
```

Part 7: Loops

With our Newfound Knowledge of if statements, we can apply this to a concept called loops. The most basic loop is a While loop as demonstrated below.

```
while(True){
    executable
}
```

Inside this while loop, we can put a conditional!

```
while(True){
    if(a == b){
    a++; // a = a+1
    a+=52/7; // a = a+52/7
    }
}
```

This also can be written as:

```
while(a == b){
    a == b;
}
```

Now, we can extrapolate this a little bit further with the following example of a for loop:

```
int x = 0;
while(x < 200){
    print(“Hello World!”);
    x++;
}
```

Which is also:

```
for(int x = 0; x < 5; x++){
    print(“Hello World!”);
}
```

Assignment:
Take in an Array(using input) and find the largest number using a loop.

Part 8: Functions

We all like being lazy, and in computer science. It is especially important to be lazy. As a result of this, we have functions to do the most basic tasks in our lives. Take for example the “print” function that pops a message up on a screen, or the rect function that draws a rectangle. These functions are a basic necessity in programming.

Take the following example:

```
def function(a,b){
    for(int i = 0; i < b-a; i++){
        print(a++);
    }
}
```

If we want to use this again and again, we would use a function to do this code for us instead of copying and pasting it. As a result, we would need to write much less

code. In for loops, we have a system called indexing which assigns each item in a set a slot number starting from 0 and continuing to any countably infinite integer.
Example:
[a,b,c,d,e,f,g,h,i,j]
 0 1 2 3 4 5 6 7 8 9
len([a,b,c,d,e,f,g,h,i,j]) = 10

(note that len(x)-1 = x[last])

Part 9: Recursion

Sometimes, you need to do the same thing again and again and you know one answer and the relation between each of the steps. We call this recursion. For recursion, there is a “base case”.

```
def reverse(in){
    if(len(in) == 1){
         return in;
    }else{
        return in[len(in)-1] + reverse(in.pop());
    }
}
```

Assignment:
Write a function that calculates any number factorial using recursion.

Part 10: Classes

A class is just a type of object. Chances are that you, the reader is a human. We can consider humans a class. A class is a category of things that share a set of common attributes. What are some common attributes of Humans?

We can say that Humans have a first and last name, an age, a height and a weight. There are other things about humans like hair, skin, and eye color. You, the human, have specific types of all these traits. For example:

First Name: John
Last Name: Doe
Age: 21
Height: 5’6
Weight: 150 lbs
Hair: Black
Skin: Brown
Eyes: Dark Brown

We can say that John Doe is an Object. An object is a physical thing that exists. While humans are just an idea, John Doe, or yourself are both physical objects that exist.

While the things in your computer aren’t physical things, we can represent physical things in our computers and we can think of the representations as a part of the actual thing.

Part 10: Inheritance

Now you may have heard your friends and family comment on how your nose looks like your mom's and that you have your eyes from your dad. You inherit traits from your parents. Just like this, classes can inherit from other classes. If we go back to our humans class, we can see that there are some things that humans share with mammals. For example, humans are warm blooded, mammals also have a size and weight, and mammals are multicellular (we know this because you can see mammals and they are not giant spherical blobs. There is an example of inheritance below:

```
class Mammal:

    function walk(){
        implementation not shown
    }
```

We know that all mammals can walk. We also know that all humans can walk, so we can

say that the walking trait is something that humans inherit from mammals. Some other examples of such traits would be giving live birth and having warm blood.

Real Code:

Right, now comes the fun part. You have an idea of code, now you just have to write it. I'd recommend Python as a good starting point because it is easy to use and widespread in the Industry.