**Introduction To Programming**

Close your eyes.

Now imagine living in a world where you had the ability to write on a piece of paper and have whatever you wrote down come into being.

You could create a world. A new species. A job. Money.

Anything.

Seems magical, right? Like spell-craft, doesn’t it? But what if I told you something like that already exists.

Open your eyes?

This thing is called CODE. When pieced together into a more complex set of instructions it is also known as a PROGRAM.

With the ability to code, you are limited only by your knowledge, equipment and creativity. By gaining the ability to program you can manifest ANYTHING you desire!

With code, a keyboard is you pencil and the computer your paper.

Arthur C Clark, a famous author once said, “Any sufficiently advanced technology is indistinguishable from magic.”

Our goal if you so choose is to make you a real world wizard, a teach you a craft that allows you to turn your thoughts into things.

Learning to code isn’t difficult. You just have to put in the time, have the desire, have the knowledge & the right teacher.

A properly coded program integrated with the right machine parts can do everything from enhancing your abilities, perform time consuming tasks you would usually do, to making well informed emotionless decisions or analysis on massive amounts of information, create simulations of real world situations and equipment for cost effective training purposes, and consequently reduce your daily costs! This will give you more time to focus on what is important! A program can do all this with fewer errors than any real person and at a much faster speed! It can perform 1000xs the work in a fraction of the time!

How?

We will get to that later.

Each idea you have and implement with technology can create jobs, manifest unimaginable things and amount of people you can help with your ideas is so much more!

**The world as we know it!**

The number of non-coding jobs is rapidly shrinking, the number of coding jobs is growing at an increasingly fast rate!

Some of the jobs markets that have been hit hard, augmented or replaced by software technology are:

1. Newspapers by blogs and websites
2. Television
3. Phone companies
4. … and many more!

Unless you know how to code, even if you spend tens on thousands on an education, your future isn’t a guarantee. The world isn’t standing still. Technology has changed things, and no matter how much you want things to go back to how things used to be, there is no going back.

I tend to use the words technology and code synonymously because most technical implementations have a program helping operate it behind the scenes.

You either move forward with the times and realize that you either have to adapt or you face reality and realise that things like Netflix are the future, or you resist like Blockbuster did, and only when it is too late, you crash.

To survive, you have to build out your ideas quickly and prove yourself.

Unless you do, you will quickly become outdated.

The results speak for themselves and the foundations of our modern day technical society speaks for itself.

Technology HAS rapidly changed all industries, and you can see it used extensively used in all disciplines including accounting, finance, economics, mathematics, the sciences and various other disciplines! So much has code been very closely entwined with the development of various industries that I believe it will become a mandatory set of skills to obtain a job in the next 100 years!

Some revolutionary changes coming up are.

I watched a documentary in which a government official spoke of some of the secret projects they were working on in the desert after he had retired. He said that the stuff they were working on in the desert was 50 years ahead of what we could currently comprehend! Not manufacture or build, but COMPREHEND!

Incredible!

To give you an examples of some of the things that have been occurring to revolutionize the world behind the scenes that you should be able to build part of or fully on your own after you have finished working with us based on what program you go through are:

1. E-commerce Stores like Amazon.

2. Financial Accounting and taxation systems.

3. Mechanical limbs for paraplegics.

4. Weather prediction software.

5. Health care databases

and much more!

So you might be asking yourself, why should I learn from this guy? He is going to help me learn, but I can do that myself, he says he is going to help me get a job, but I can get one myself, and there are a ton of other programs out there and on the internet!

SURE YOU CAN FIGURE THIS OUT YOURSELF!

But it will take you much longer, and require much more sweat and blood, and most of the existing programs are built by people that learn’t on their own or have only worked at one or two companies. There are ton’s of ways in order to build something, and building something with the wrong approach will make your life much harder than it needs to be.

I’ve been through it.

So here is a little bit about me.

I obtained a Bsc in Computer Engineer at the University of Ottawa around 15 years ago and have held positions developing hardware, database development, and web design at numerous large and small companies at numerous points in my life.

I’ve participated in many rebuilds of software at companies as well as built frameworks and many websites in various technologies for my own purposes.

I’ve also completed several technical bootcamps and courses on Udemy, Pleuralsight, Infosec, Coding Dojo, Coursea, and many more.

What I found in many cases, as I went from job to job and company to company is that often times people would have the same inefficiencies! Furthermore, more often than not the tutorials and lessons were taught by people that were inexperienced or had only built things in one language!

If you join my team, I guarentee to bring you to a level where you become the best of the best! If you put in the work and prove your willing to work as hard as me or harder you WILL SUCCEED!

Outside of coding, I enjoy watching movies, art, music, going to the gym, kickboxing, writing (I’ve written a few books), volunteering, traveling and helping others.

I believe in the future coding will similar to what reading was 200 years ago. Two hundred years ago, less than 12% the population could read! Today, it is closer to 84%!

----

There is a crime being committed in America. Many of us are unaware of how we are being screwed over, because, they often pass it off as having our best interests in mind.

The education system as it is today brainwashes us, catagorizes us, and pits us against each other.

It tells us that grades or rankings created by teachers is what defines us for the rest of our lives.

When we graduate, we are cast into a role. You'll often hear people brag about how they went to Harvard, how they have a PhD, were straight A students, were once the star athlete of their football team, or other past leaps and sucesses that they've had in the past. This is great!

But fast forward a couple of years, they aren't doing anything with their lives and have stagnated. They have stopped trying!

Many people end up miserable and working a job they hate. A Gallup study said that as much as 85% of people hate their job!

I don't want this to be you.

When you walk away, having trained with us, I want to you to have what you need to be successful, be happier, more confident, more positive and more productive member of society.

But most of all be happy.

I also want to make sure you have the right mindset to endure all our future struggles.

When I started my Engineering Degree the teachers gave a pep talk and said that 50% of the people in that room wouldn't be there at the end to finish the degree. The teachers seemed proud of this, while earning a paycheck provided by naive student's monetary funds obtained from bank and government loans or their parents. It was clear that they didn't care about it as people.

Schools today teach you, take tons of money, toss you out and you never hear from them again. Most teachers don't care about you unless they are held accountable.

See this. Don't be blind to this.

So consequently, I want to tell you somethings about life that I wish I had realised earlier.

1) The first lesson I want you to take to heart is about failure is.

Failure isn't a bad thing. As long as you give it your all you will benefit and learn from it. It sucks, it hurts your ego, but we are all going to screw up at some point in our lives. Whether it is doing bad on a test, a girl, a job, getting laid off, a failed project or a business venture. You are going to face it and you are going to feel like crap about it.

Stuff happens.

But the key to happiness is getting up and trying again, or trying something different and not being discouraged. The more you fail trying different things while giving it your all and preparing, the closer you are to hittnig the jackpot with a success. You and others learn from your failures! Change what you did wrong, talk to the right people, and figure it out.

But if you are going to fail, fail big. Make sure you win at something you really want.

We are taught to believe that life is a sprint and one failure is the end of the world. The reality is that life isn't a sprint, it is a marathon filled with hills and just because at one point you are struggling, doesn't mean that you can't change something and get better. You aren't always going to enjoy it, reality is you will enjoy bits and pieces of it.

You will find some parts of the material hard and some easy. That is just how it is. Don't let how others are performing discourage you. Ask questions and don't criticize. Learn from each other, the internet and me. Make your mistakes now.

The more work you put in now while learning, the easier things will be later. In this class and in life.

We as humans learn, develop and become stronger with each struggle we face and overcome. The harder we struggle, regardless of whether we fail or succeed, the stronger we get. Mentally and physically.

2) Second lesson is life is easier when you have a team backing you.

You may get pretty far on your own if you are gifted, however, it is going to be a LOT harder and often times will take a lot longer.

A untalented team that works well together will defeat the most talented of individuals.

A talented team backing a talented individual will make that individual reach divinity!

3) The third lesson I want to tell you is you won't always have support on your journeys.

As you learn, people may criticize your attempts, put you down, or shower you with negativity. Maybe it will be from family, coworkers, bosses, friends. Perhaps, based on judgments of you they made based on your behavior from the past. Often times, at the most unexpected times, random people will say things that make you doubt yourself. They will point out your mistakes, sabotage you and point out your flaws out of jealousy or arrogance. But the reality is that this is life. No one is going to hold your hand and encourage you, and if they do, be worried, because it is making you weaker. Take the feedback and learn from it to become better at your craft, especially, if it is from someone that doesn't control your employment or salary. But if they are super negative and toxic cut them off at least during this class. Be positive!

Sometimes, getting that team is difficult until you prove yourself until you get the ball rolling and prove yourself!

4) The fourth take away is give it your all. BE POSITIVE and give it your all and be engaged.

I can't do the learning for you. I can't ask you to do extra to become better. Being enganged will help you learn better!

The beauty about coding is that a University Degree, PhD or job experience isn't needed to build something amazing! No matter how slow you were learning the material, no one will care in the future as long as you can do the job well! So the guy that learn't everything in a day will be seen no differently from the person that learn't it in a month if you do things equally as well in the job place.

You need discipline to build and you don't have to build it perfectly as long as you get it working!

The beauty about knowledge is that there is no short cuts to it. Sure, some people may learn more quickly than others, absorb it better, but they are still going to have to put in the grunt work.

Make the mistakes to learn the details!

Do the research!

Ask questions!

Help each other and work hard!

Everyone I know is at a different point in their life. Don't let your self doubt, negativity, worries and personal problems weigh you down. If anything is on your mind feel free to reach out to us whether it is about the class or something else.

5) The fifth and final lesson, prior to starting is this. Don't get distracted by shiny objects and remember your goals.

There will be various other technologies, arguments. Maybe a beautiful girl, a investment opportunity, etc ...

but if you don't focus and your attention is everywhere, regardless of how talented you are, your chances of success are innumerably lower.

----

Foundational concepts:

----

Are you ready to learn!?

Okay, so now lets get into the meat. Start learning.

Prior to starting, I want to make sure that everyone is at the same point, and you have the right mindset starting. Programming is meant to be fun! So prior to learning anything, we will be playing a quick game! Remember there is no right or wrong!

So for the first section, in which we will introduce you to creating simple program and creating characters for a game we will compete in, we are just asking for 200$s. This will go towards marketing you for job positions if you choose to continue. The actual course fees will come later. This should weed out the people who are not serious. I’d would do it for free, however, what is free isn’t appreciated. I am going to have to hire help since marketing isn’t my fortee in a ever increasingly competitive market and paying systems that the course work. Creating a better future for you requires a good team backing you, and effort from all parties involved! The harder you work now, the easier your work life will be later!

----

**Introduction to Programming Languages**

You will often hear about different programming languages. Thousands have been widely used, grown to be hugely popular and come down crashing over night or been slowly phased out of use. Some endured. This cycle often repeats in the programming industry.

So what is the difference? Why choose one programming language over another?

Often time different programming languages are good for different things.

I have seen it broken up in different ways, but I like to break it up into the following categories initially for teaching purposes. These are simply:

1. Low level languages

2. Middle level languages

3. Higher level languages

**Low level languages** are languages are used for interacting directly with hardware. These include Assembly, or machine code. These are a set of instructions that you write to interact with hardware components directly. They are highly dependent on the setup of your computer’s hardware and send instructions directly to each hardware component. In the real world, there would be no reason to create programs with these languages unless you want to write a program that:

1. Utilizes very little power (higher level languages tend to use more electricity). Some scenarios this might be necessary in is:
   1. Mars rovers
   2. Heart pace makers
2. You need it to run extremely fast. Yet in this case, it would make more sense to simply convert the code into a hardware implementation because hardware will out perform the low level language (although a piece of code would be cheaper in the short term). A scenario this might be used in is:
   1. Algorithm calculations

A short coming of low level languages is that they will only work with one hardware configuration and not be portable to hardware setup in a different way.

**Middle level languages** are languages built from low level languages. For example, C is a programming language often used in robotics, and was coded in Assembly. These languages were initially made popular because they would reduce the amount of code you would have to write exponentially compared to the utilization of lower level languages. Furthermore, they were portable from one hardware configuration to another in many cases. They were often used for the same reason as low level languages like Assembly and Machine languages and still are! But there are still several complexities you have to deal with when working with a Middle level language. In essence, you have to know more when working for low and middle level languages than with higher level languages because if you use them improperly you can accidentally expose your system to security issues, memories issues or even write highly inefficient code. Furthermore, while the code isn’t hardware specific, the code you write in these languages tends to be operating system (OS) specific. So for example, a program written in C will either work on Linux, Windows or Mac OS, instead of all.

Something called a compiler is used to convert the code you write into Machine language readable by each OS! So at the end of the day, in my opinion, its main purpose is to write code exponentially faster!

So there is clearly room for improvement! This is where the higher level languages come into play!

**Higher level languages** are languages built from Middle level languages. Examples include C# and Java. These languages take care of an everything middle level languages did and more! Furthermore, they run on top of another program called a Virtual Machine. A virtual machine can be installed on any operating system and allows for programs that are written in higher level languages to run on any operating system by interacting with it, rather than the operating system!

Furthermore, these languages often take care of many of the coding complexities that the middle level languages didn’t, and at times don’t even expose them. Like:

1. Memory management and
2. Security loop holes (buffer overflow, insecure operations etc …)
3. Bad coding practices

If you are going to write a program and you are not limited by hardware speed, memory, or power usage, I’d suggest utilizing higher level languages in most cases! Furthermore, they allow for you to code exponentially faster!

So now that you know that, lets delve into the program paths you can choose from:

1. Database Engineering
2. Web Development
3. Web Development & Web Security (Coming soon)

If you have difficulty understanding these concepts, don’t worry, we will revisit them later. However, feel free to ask questions! The important stuff comes next!

---

**Foundations Of Programming**

Let me say this early. The more you know & understand and better you program, the less work you will have later on! So as you progress, common questions asked like:

1. Why build a website from scratch rather than using Wix or Wordpress?
2. Why are best practices important?
3. How long will it take for me to
4. How programming structures differ from company to company

The best type of understanding comes through doing. Consequently, in your first introduction to coding we will be writing a code for a game called Battle Robots after a brief introduction to programming concepts. What you learn here will be used in all the other courses. Don’t be afraid to make mistakes, don’t try and do things perfectly, just have fun and give it a go!

The game we will be using to learn how to code is: Robocode!

You can download it at the following link: https://robocode.sourceforge.io/

This game is very close to my heart. When I was 8 years old, that is over 25 years ago, my cousin introduced me to this game, it is one of the things that first got me into programming and I hope it will do the same for you!

Programming is fun and is a different way of thinking. Its a magical skill and I want you to see it as such.

So go to robotocode.sourceforge.io, download the code and install the jvm and game as shown in the video I am about to play. Then