GOOGLE IT SUPPORT SPECIALIZATION
COURSE 1: TECHNICAL SUPPORT FUNDAMENTALS
01 INTRODUCTION TO IT

Table des matières

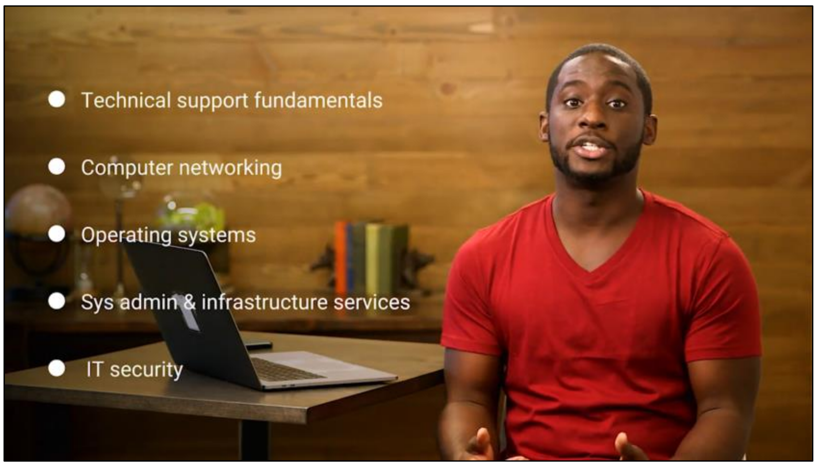
1 Introduction to IT

1.1 Introduction to IT support

1.1.1 Program introduction

There's a big problem in the world right now. There are hundreds of thousands of IT support jobs just waiting for skilled candidates to fill them. They're available at this very moment and there are companies large and small that really want to hire motivated people. With technology seeping into nearly every aspect of business is growing by the second but that's just half the story. There are lots of people around the world like you, who are looking for a flexible way to learn the skills necessary to get that entry level IT support role. There might be a few obstacles in the way, maybe you don't have a university degree, or the access, or flexibility to take in person training's, or maybe the cost is just too high. Whatever the reason, you're looking for an accessible hands on way to learn the skills that companies are hiring for. Google and Coursera are thrilled to welcome you to the IT Support Professional Certificate Program. This program is designed to give you the foundational skills and confidence required for an entry level IT support role, whether it's doing in person support, or remote support or both. What's really special about this program is that learners get a hands on experience, through a dynamic mix of labs and other interactive exercises. Just like what you'd encounter in a help desk role. This curriculum is designed to get you job ready. But we're taking it one step further, when you complete this program you'll have your opportunity to share your information with Google, Bing of America and other top employers looking to hire entry level IT support professionals. This program has been designed to teach anyone who's interested in learning the foundational skills and IT support. Doesn't matter if you've been tinkering with IT on your own or you're completely new to the field, we'll bring the content developed entirely by Googlers and you bring the passion and motivation to learn.

Here's how we're going to get there, this program is rooted in the belief that a strong foundation and IT support can serve as a launch pad to a meaningful career in IT And so we've designed six industry relevant courses. Technical Support fundamentals, computer networking, operating systems, systems administration and IT infrastructure services, IT automation and IT Security.


Technical support fundamentals
Computer networking
Operating systems
Sys admin & infrastructure services
IT security

If you dedicate around eight to 10 hours a week to the courses we anticipate that you'll complete the certificate in about eight months and learning this stuff won't be like your typical classroom experience. You can move through the material at your own pace, skip content that you might already know or review the lessons again if you need a refresher. It's choose your own adventure experience, plus we think that the lint is a strong signal to employers that you have the grit and persistence it takes to succeed in an ever changing field like IT. Another really cool part about this program is that it's been created entirely by real world pros who have a strong background in IT support. Their work in IT fields like operations engineering, security, site reliability engineering and systems administration. They know this content because they live it every day along the way you're going to hear from Googlers with unique backgrounds and perspectives, they'll share how their foundation in IT support served as a jumping off point for their careers. They also give you a glimpse into the day to day work along with tips on how to approach IT support interviews, they'll even share personal obstacles that they've overcome in inspiring ways. They're excited to go on this journey with you as you invest in your future by achieving an end of program certificate.

Last but not least, we gathered a truly amazing group of course instructors for you to learn from. They've all worked in IT support at Google and are excited to guide you through the content step by step. Ready to meet them? They're all really excited to meet you.

My name is Kevin Limehouse and I'm a support specialist for platforms and DoubleClick. I'm going to present the history of computing in course one.

I'm Victor Escobedo, and I'm a corporate operations engineer. We'll meet in the lessons on the Internet in the first course of technical support fundamentals, then I'll be your instructor for course two the bits and bytes of computer networking.

Hey I'm Cindy Quach and I work in site reliability engineering. I'll be teaching you about operating systems in course one and then take you through a much deeper dive in OS's in course three, operating systems and you becoming a power user. My name is Devan Sri-Tharan and I work in corporate operations engineering at Google. We're going to cover all the hardware and even build a computing course one with me again when I teach course four, systems administration in IT infrastructure services.

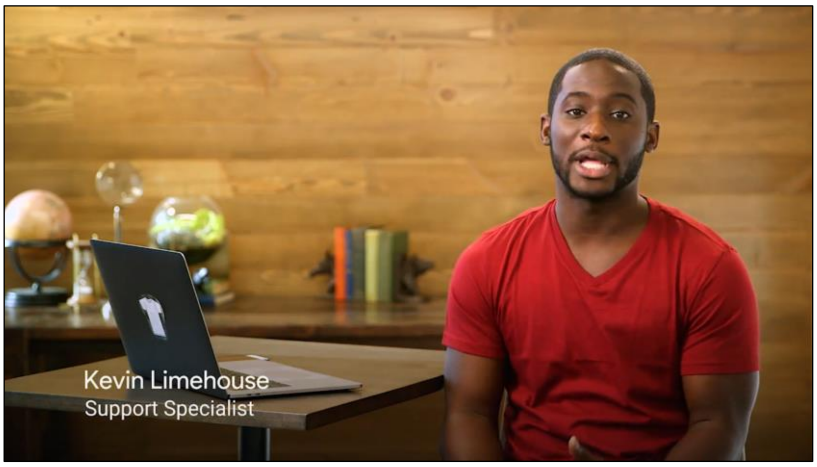
Hey everyone, my name is Phelan Vendeville and I'm a systems engineer on Google's site reliability team. I'm excited to teach the software lessons to you in course one, I'll also be your instructor for course five, automation. It's not that scary. Really it isn't.

Hi, my name is Gian Spicuzza and I'm a program manager in Android Security. I'm going to teach you about the history and the impact of the Internet in course one, and then I'll be your instructor for the last course of this program, IT security, defense against the digital dark arts.

Hi, my name is Marti Clark and I'm a manager with Google's internal IT support team. I'll be teaching you about troubleshooting, customer service and documentation in course one.

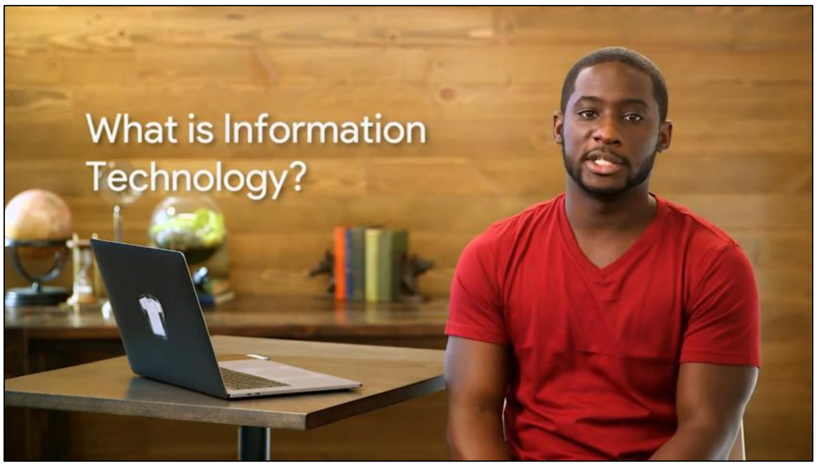
1.1.2 What is IT

Welcome to course one, Technical Support Fundamentals. My name is Kevin Limehouse and I work as a support specialist for platforms building DoubleClick at Google.

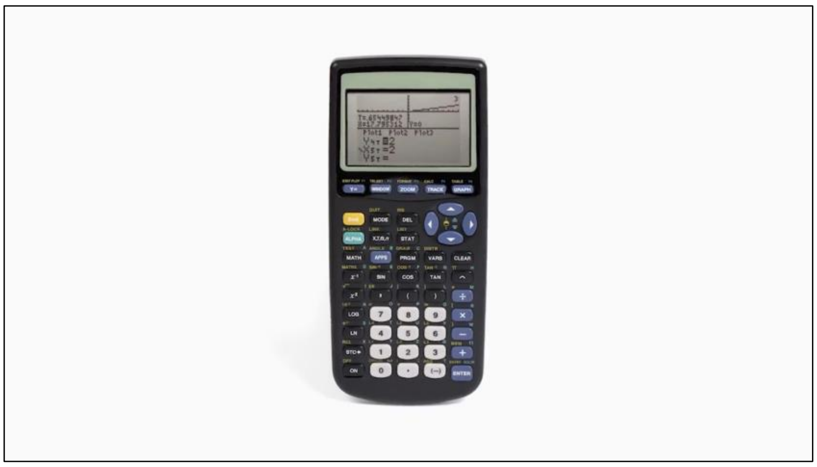


Looking back, I can trace from my passion for IT began, to an actual moment when I was eight years old. My parents were about to throw away our old busted computer, but I managed to convince my mom to let me keep it. I remember the moment when I slowly started disassembling it, kept digging deeper and deeper unscrewing every little piece I can get my hands on and I was hooked. By the time I was 12 or 13 years old, I became the de facto IT support for my entire family and that's no small feat considering, I have 11 aunts and uncles and over 35 cousins. My parents both grew up in very small rural towns in South Carolina. Growing up in the Jim Crow south through the mid 1950s and 1960s, they were taught at an early age that one of the better methods to get ahead was through education. This lesson was instilled in me and my sister and I ended up going to university to study computer science. I graduate at school right at the end of the 2007, 2009 recession, but thankfully I secured a job at Google in IT support, where I work with users, solving their issues and supporting the IT inventory. And now I've been working in IT for seven years and my current role as a support specialist, I provide technical and billing support to the Google sales teams which involves everything from troubleshooting to creating forms or editing automation scripts. Now you know a little bit about me.

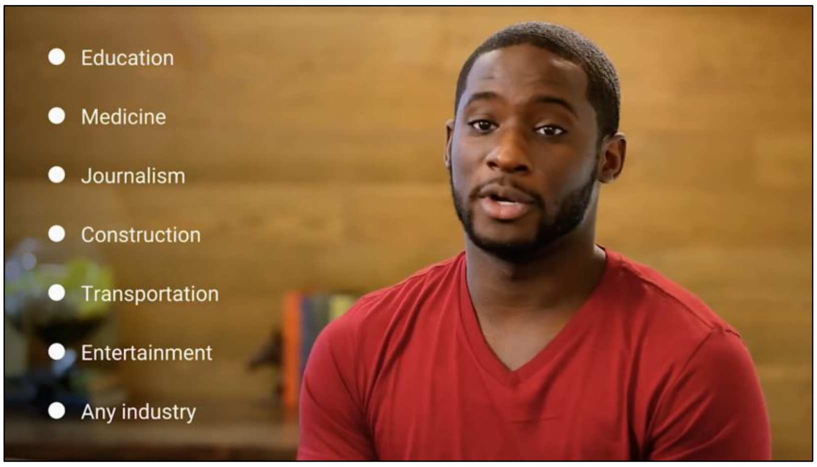
Let's start from the beginning. What is information technology?



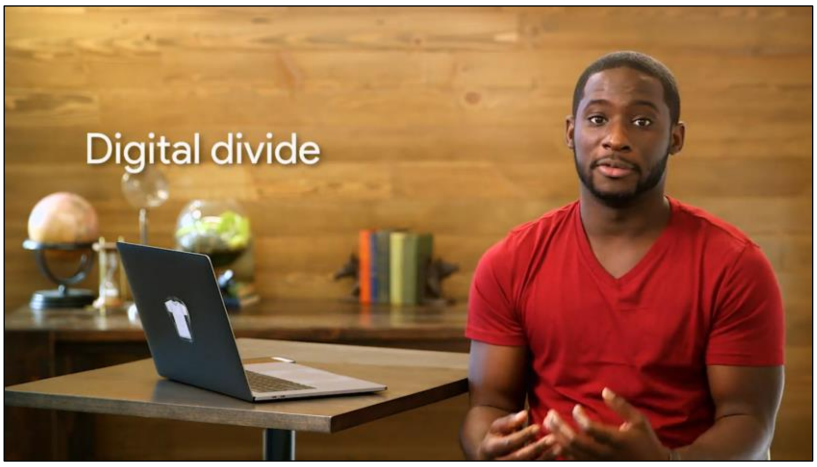
Information technology has completely transformed your life in ways that you may not even realize. Thanks to IT, we can communicate massive amounts of information to people and organizations across the world, in the blink of an eye. Computers power everything from calculators, to medical equipment, to complex satellite systems and the trading desk of Wall Street.



They are powerful and invaluable tools to help people get their work done and enable us to connect with one another. So what exactly is information technology? IT is essentially the use of digital technology, like computers and the internet, to store and process data into useful information. The IT industry refers to the entire scope of all the jobs and resources that are related to computing technologies within society, and there are a lot of different types of jobs in this field, from network engineers who ensure computers can communicate with each other, to hardware technicians who replace and repair components, to desktop support personnel who make sure that end users can use their software properly. But IT isn't just about building computers and using the Internet, it's really about the people. That's the heart and soul of IT support work. What good is technology or information if people can't use technology or make sense of the information? IT helps people solve meaningful problems by using technology, which is why you'll see its influences in education, medicine, journalism, construction, transportation, entertainment, or really any industry on the planet.



IT is about changing the world through the ways we collaborate, share and create together. IT has become such a vital tool in modern society that people and organizations who don't have access to IT are at a disadvantage. IT skills are becoming necessary for day to day living, like finding a job, getting an education and looking up your health information. Maybe you're from a community where there wasn't any internet, or you couldn't afford a superfast computer and had to use one at your school or library instead. There are many social and economic reasons why some people have digital literacy skills and other people do not. There's growing skills gap is known as the digital divide.



People without digital literacy skills are falling behind. But people like you are the real solution to bridging that digital divide. Overcoming the digital divide, not only involves confronting and understanding the combination of socio economic factors that shape our experience, but also helping others confront and understand those experiences. By getting into IT, you'll help serve those in your communities or organizations and maybe even inspire a new generation of IT pioneers. When I think about solving the digital divide, I can't help but think of all the opportunities and breakthroughs that folks from diverse backgrounds and perspectives in the industry can bring. By bringing more people of color, more women, more ethnically diverse people into the IT fields, we're bound to see unique new ideas and products that we haven't even begun to imagine that benefits everybody.

1.1.3 What does an IT support specialist do

So what's the day-to-day work of someone in IT support like? Well, it varies a ton, based on whether you're doing in personal or remote, support and at a small business or a large enterprise company. And there's really no such thing as day-to-day work, since the puzzles and challenges are always new and interesting. But in general, an IT support specialist makes sure that an organization's technological equipment is running smoothly. This includes managing, installing, maintaining, troubleshooting and configuring office and computing equipment. This program is designed to prepare you for an entry level role in IT help desk support. You'll learn how to setup a user's desktop or workstation, how to install the computer applications that people use the most. You'll learn how to fix a problem or troubleshoot when something goes wrong and how to put practices in place to prevent similar problems from happening again. Not only will you learn the technical aspects of troubleshooting a problem, you'll also learn how to communicate with users in order to best assist them. We'll also show you how to set up a network from scratch to connect to the Internet. Teach you a thing or two about automation and scripting and teach you about how to implement security to make sure your systems are safe from hackers and other risk. For me, my favorite part of IT support is the problem solving aspect. I love to exercise my creativity to spin up a solution to a user's issue. Being an IT journalist also gave me the flexibility to learn and practice so many different skills and eventually determined where I want to focus my career. Plus, when things go wrong or you fail at something in IT, you can take the feedback from those mistakes and be better equipped to tackle them the next time around. Using failure as a feedback, is an important skill both in IT and in life. For me, that's why I was so attracted to the IT field. I love the process of problem solving and constantly stretching myself to learn and grow. There's also never been more opportunity to get into the IT industry than now. Not only is the field of IT incredibly diverse, but job prospects are also booming. It's projected that IT jobs in the US alone will grow 12% in the next decade. That's higher than the average for all other occupations. So what does this all mean? There are thousands of companies around the world searching for IT professionals like you. So the main gist is that IT is totally awesome and full of opportunity, and we're so excited that you're here. So let's dive right in.

1.1.4 Course introduction

On July 20th, 1969, one of the most phenomenal events made his way into the history books. When the Apollo 11 completed its historic mission to the moon. Although the most brilliant minds helped to make sure that the Eagle had landed, computers also played a significant role. The guidance system that navigated the spacecraft was one of the earliest forms of modern computing. That same computer, the one that helped America's lunar dreams become a reality, took up the space of an entire room and had one-10,000th the computing power of the thing that almost everyone of you carry in your pockets today, a smartphone. Computer hardware and software have had such a dramatic evolution that what was once only used to power rockets, now shapes the entire way our world functions. Think about your day, did you grab a snack? Turn on your TV? Take a driving in your car? Computers were along for the ride literally, computers are everywhere. So, here's the rundown. By the end of this course, you'll understand how computers work and get a grasp of the building blocks of IT. We're going to cover the basics of how computer hardware performs calculations, and we're going to actually build a computer from the ground up. We'll look at how operating systems control and interact with hardware, we'll take a look at the Internet and get a better understanding of how computers talk to each other. We'll also spend time learning about how applications and programs tie all of this together, and let humans interact with these systems. Finally, we'll cover important lessons on problem-solving with computers, and cover the communication skills that are so critical when interacting with others in IT. Whether you're looking for a job in the IT industry or you just want to learn higher laptop connects to the Internet, understanding how computers work at every level can help you in your day-to-day life and in the workplace. But first, let's take a step way back, back to where all begin, even before the Apollo 11 mission touchdown so you can understand how and why we use computers today.

1.2 History of computing

1.2.1 From abacus to analytical engine

When you hear the word computer, maybe you think of something like a beefy gaming desktop with flashing lights, or maybe you think of a slim and sleek laptop. These fancy devices aren't what people had in mind when computers were first created. To put it simply, a computer is a device that stores and processes data by performing calculations. Before we had actual computer devices, the term computer was used to refer to someone who actually did the calculation. You're probably thinking that's crazy talk. A computer lets me check social media, browse the Internet, design graphics, how can it possibly just perform calculations? Well, friends, in this course, we'll be learning how computer calculations are baked into applications, social media, games et cetera, all the things that you use every day. But to kick things off, we'll learn about the journey computers took from the earliest known forms of computing into the devices that you know and love today. In the world of technology, and if I'm getting really philosophical, in life, it is important to know where we've been in order to understand where we are and where we are going. Historical context can help you understand why things work the way they do today. Have you ever wondered why the alphabet isn't laid out in order on your keyboard? The keyboard layout that most of the world uses today is the qwerty layout, distinguished by the Q, W, E, R, T, and Y keys in the top row of the keyboard. The most common letters that you type aren't found on the home row, where your fingers hit the most. But why? There are many stories that claim to answer this question. Some say it was developed a slow down typist so they wouldn't jam old mechanical typewriters. Others claim it was meant to resolve problem for telegraph operators. One thing is for sure, the keyboard layout that millions of people use today isn't the most effective one. Different keyboard layouts have even been created to try and make typing more efficient. Now that we're starting to live in a mobile-centric world with our smartphones, the landscape for keyboards may change completely. My typing fingers are crossed. In the technology industry, having a little context can go a long way to making sense of the concepts you'll encounter. By the end of this lesson, you'll be able to identify some of the most major advances in the early history of computers. Do you know what an abacus is? It looks like a wooden toy that a child would play with, but it's actually one of the earliest known computers. It was invented in 500 BC to count large numbers. While we have calculators like the old reliable TI-89s or the ones in our computers, abacuses actually are still used today. Over the centuries, humans built more advanced counting tools but they still required a human to manually perform the calculations. The first major step forward was the invention of the mechanical calculator in the 17th by Blaise Pascal. This device used a series of gears and levers to perform calculations for the user automatically. While it was limited to addition, subtraction, multiplication and division for pretty small numbers, it paved the way for more complex machines. The fundamental operations of the mechanical calculator were later applied to the textile industry. Before we had streamlined manufacturing, looms were used to weave yarn into fabric. If you wanted to design patterns on your fabric, that took an incredible amount of manual work. In the 1800s, a man by the name of Joseph Jacquard invented a programmable loom. These looms took a sequence of cards with holes in them. When the loom encountered a hole, it would hook the thread underneath it. If it didn't encounter a hole, the hook wouldn't thread anything. Eventually this spun up a design pattern on the fabric. These cards were known as punch cards. And while Mr. Jacquard reinvented the textile industry, he probably didn't realize that his invention would shaped the world of computing and the world itself today. Pretty epic Mr. Jacquard, pretty epic. Let's fast forward a few decades and meet a man by the name of Charles Babbage. Babbage was a gifted engineer who developed a series of machines that are now known as the greatest breakthrough on our way to the modern computer. He built what was called a difference engine. It was a very sophisticated version of some of the mechanical calculators we were just talking about. It could perform fairly complicated mathematical operations but not much else. Babbage's follow up to the difference engine was a machine he called the Analytical Engine. He was inspired by Jacquard's use of punch cards to automatically perform calculations instead of manually entering them by hand. Babbage used punch cards in his Analytical engine to allow people to predefine a series of calculations they wanted to perform. As impressive as this achievement was, the Analytical engine was still just a very advanced mechanical calculator. It took the powerful insights of a mathematician named Ada Lovelace to realize the true potential of the analytical engine. She was the first person to recognize that the machine could be used for more than pure calculations. She developed the first algorithm for the engine. It was the very first example of computer programming. An algorithm is just a series of steps that solves specific problems. Because of Lovelace's discovery that algorithms could be programmed into the Analytical engine, it became the very first general purpose computing machine in history, and a great example that women have had some of the most valuable minds in technology since the 1800s. We've covered a lot of ground already, learning about how primitive counting devices like the abacus evolved into huge complex devices like the Analytical engine, proof that there was life before social media. In the next video, we'll learn about how these mechanical machines made the leap into modern computing.

1.2.2 The path to modern computers

Welcome back. In this video, we'll be learning how huge devices like the Analytical Engine grew, I mean, shrunk into the computing devices that we use today. The development of computing has been steadily growing since the invention of the Analytical Engine but didn't make a huge leap forward until World War II. Back then, research into computing was super expensive, electronic components were large and you needed lots of them to compute anything of value. This also meant that computers took up a ton of space and many efforts were underfunded and unable to make headway. When the war broke out, governments started pouring money and resources into computing research. They wanted to help develop technologies that would give them advantages over other countries, lots of efforts were spun up and advancements were made in fields like cryptography. Cryptography is the art of writing and solving codes. During the war, computers were used to process secret messages from enemies faster than a human could ever hope to do. Today, the role cryptography plays in secure communication is a critical part of computer security which we'll learn more about in a later course. For now, let's look at how computers started to make dramatic impact on society. First up is Alan Turing, an English mathematician and now famous computer scientist. He helped develop the top-secret Enigma machine which helped Allied Forces decode Axis messages during World War II. The Enigma machine is just one of the examples of how governments started to recognize the potential of computation. After the war, companies like IBM, Hewlett-Packard, and others were advancing their technologies into the academic, business, and government realms. Lots of technological advancements and computing were made in the 20th century thanks to direct interest from governments, scientists, and companies left over from World War II. These organizations invented new methods to store data in computers which fueled the growth of computational power. Consider this, until the 1950s punch cards were a popular way to store data. Operators would have decks of ordered punch cards that were used for data processing. If they dropped the deck by accident and the cards got out of order, it was almost impossible to get them sorted again. There were obviously some limitations to punch cards, but thanks to new technological innovations like magnetic tape and its counterparts, people began to store more data on more reliable media. A magnetic tape worked by magnetizing data onto a tape. Back in the 1970s and 80s, people used to listen to music on vinyl records or cassette tapes. These relics are examples of how magnetic tapes can store information and run that information from a machine. This left stacks and stacks of punch cards to collect dust while their new magnetic tape counterparts began to revolutionize the industry. I wasn't joking when I said early computers took up a lot of space. They had huge machines to read data and racks of vacuum tubes that help move that data. Vacuum tubes control the electricity voltages and all sorts of electronic equipment like televisions and radios, but these specific vacuum tubes were bulky and broke all the time. Imagine what the work of an I.T. support specialist was like in those early days of computing. The job description might have included crawling around inside huge machines filled with dust and creepy crawly things, or replacing vacuum tubes and swapping out those punch cards. In those days, doing some debugging might have taken on a more literal meaning. Renowned computer scientist Admiral Grace Hopper had a favorite story involving some engineers working on the Harvard Mark II computer. They were trying to figure out the source of the problems in a relay. After doing some investigating, they discovered the source of their trouble was a moth, a literal bug in the computer. The ENIAC was one of the earliest forms of general purpose computers. It was a wall-to-wall convolution of massive electronic components and wires. It had 17,000 vacuum tubes and took up about 1,800 square feet of floor space. Imagine if you had to work with that scale of equipment today. I wouldn't want to share an office with 1,800 square feet of machinery. Eventually, the industry started using transistors to control electricity voltages. This is now a fundamental component of all electronic devices. Transistors perform almost the same functions as vacuum tubes but they are more compact and more efficient. You can easily have billions of transistors in a small computer chip today. Throughout the decades, more and more advancements were made. The very first compiler was invented by Admiral Grace Hopper. Compilers made it possible to translate human language via a programming language into machine code. In case you didn't totally catch that, we'll talk more about compilers later in this course. The big takeaway is that this advancement was a huge milestone in computing that led to where we are today. Now, learning programming languages is accessible for almost anyone anywhere. We no longer have to learn how to write machine code in ones and zeros. You get to see these languages in action in future lessons where you'll write some code yourself. Side note, if the thought of that scares you, don't worry, we'll help you every step of the way. But for now, let's get back to the evolution of computers. Eventually, the industry gave way to the first hard disk drives and microprocessors. Then, programming language started becoming the predominant way for engineers to develop computer software. Computers were getting smaller and smaller, thanks to advancements in electronic components. Instead of filling up entire rooms like ENIAC, they were getting small enough to fit on tabletops. The Xerox Alto was the first computer that resembled the computers we're familiar with now. It was also the first computer to implement a graphical user interface that used icons, a mouse, and a window. Some of you may remember that the sheer size and cost of historical computers made it almost impossible for an average family to own one. Instead, they were usually found in military and university research facilities. When companies like Xerox started building machines at a relatively affordable price and at a smaller form factor, the consumer age of computing began. Then in the 1970s, a young engineer named Steve Wozniak invented the Apple I, a single-board computer MIT for hobbyists. With his friend Steve Jobs, they created a company called Apple Computer. Their follow up to the Apple I, the Apple II, was ready for the average consumer to use. The Apple II was a phenomenal success, selling for nearly two decades and giving a new generation of people access to personal computers. For the first time, computers became affordable for the middle class and helped bring computing technology into both the home and office. In the 1980s, IBM introduced its personal computer. It was released with a primitive version of an operating system called MS DOS or Microsoft Disk Operating System. Side note, modern operating systems don't just have text anymore, they have beautiful icons, words, and images like what we see on our smartphones. It's incredible how far we've come from the first operating system to the operating systems we use today. Back to IBM's PC, it was widely adopted and made more accessible to consumers, thanks to a partnership with Microsoft. Microsoft, founded by Bill Gates, eventually created Microsoft Windows. For decades it was the preferred operating system in the workplace and dominated the computing industry because it could be run on any compatible hardware. With more computers in the workplace, the dependence on I.T. rose and so did the demand for skilled workers who could support that technology. Not only were personal computers entering the household for the first time, but a new type of computing was emerging: video games. During the 1970s and 80s, coin-operated entertainment machine called arcades became more and more popular. A company called Atari developed one of the first coin-operated arcade games in 1972 called Pong. Pong was such a sensation that people were standing in lines at bars and rec centers for hours at a time to play. Entertainment computers like Pong launch the video game era. Eventually, Atari went on to launch the video computer system which help bring personal video consoles into the home. Video games have contributed to the evolution of computers in a very real way, tell that to the next person who dismisses them as a toy. Video game show people that computers didn't always have to be all work and no play, they were a great source of entertainment too. This was an important milestone for the computing industry, since at that time, computers were primarily used in the workplace or at research institutions. With huge players in the market like Apple Macintosh and Microsoft Windows taking over the operating systems space, a programmer by the name of Richard Stallman started developing a free Unix-like operating system. Unix was an operating system developed by Ken Thompson and Dennis Ritchie, but it wasn't cheap and wasn't available to everyone. Stallman created an OS that he called GNU. It was meant to be free to use with similar functionality to Unix. Unlike Windows or Macintosh, GNU wasn't owned by a single company, its code was open source which meant that anyone could modify and share it. GNU didn't evolve into a full operating system, but it set a foundation for the formation of one of the largest open source operating system, Linux, which was created by Linus Torvalds. We'll get into the technical details of Linux later in this course, but just know that it's a major player in today's operating systems. As an I.T. support specialist, it is very likely that you'll work with an open source software. You might already be using one like the internet browser Mozilla Firefox. By the early 90s, computers started getting even smaller, then a real game changer made its way into the scene: PDAs or personal digital assistants, which allows computing to go mobile. These mobile devices included portable media players, word processors, email clients, Internet browsers, and more all in one handy handheld device. In the late 1990s, Nokia introduced a PDA with mobile phone functionality. This ignited an industry of pocketable computers or as we know them today, smartphones. In mere decades, we went from having computers that weigh tons and took up entire rooms to having powerful computers that fit in our pockets. It's almost unbelievable, and it's just the beginning. If you're stepping into the I.T. industry, it's essential that you understand how to support the growing need of this ever-changing technology. Computer support 50 years ago consisted of changing vacuum tubes and stacking punch cards, things that no longer exist in today's I.T. world. While computers evolve in both complexity and prevalence, so did knowledge required to support and maintain them. In 10 years, I.T. support could require working through virtual reality lenses, you never know. Who knows what the future holds? But right now, it is an exciting time to be at the forefront of this industry. Now that we've run down where computers came from and how they've evolved over the decades, let's get a better grasp on how computers actually work.

1.2.3 Kevin career

I think I realized I could pursue this like IT support as a career my freshman year of high school. So, I took an intro to computer applications class and that's when you just learn about like a lot of the like the very, very basics of computers. And our teacher always talked about how this is where the world is going. This is in 2001 and getting this foundational knowledge at a young age of 14, 15 is like going to help you a lot and you're moving into a college and leaving school and trying to get an actual job. Well fortunately enough, my first job was working with Google. I started here maybe a month after graduating and I was like doing like very, very entry level low level tech support. One of the best memories, one like the best accomplishments I think I have from my IT support job was training some of the new people in the program that I was a part of. So, I guess, it's like a win knowing that not only myself who eventually left the program and went on to other things, people that I brought on, helped train, helped teach, have moved on and done better things.

1.3 Digital logic

1.3.1 Computer language

Remember when I said that a computer is a device that stores and processes data by performing calculations? Whether you're creating an artificial intelligence that can beat humans at chess or something more simple, like running a video game, the more computing power you have access to, the more you can accomplish. By the end of this lesson, you'll understand what a computer calculates, and how. Let's look at this simple math problem. 0 +1 equals what? It only takes a moment to come up with the answer 1, but imagine that you needed to do 100 calculations that were this simple. You could do it, and if you were careful, you might not make any mistakes. Well, what if you needed to do 1,000 of these calculations? How about 1 million? How about 1 billion? This is exactly what a computer does. A computer simply compares 1s and 0s, but millions or billions of times per second. Wowza! The communication that a computer uses is referred to as binary system, also known as base-2 numeral system. This means that it only talks in 1s and 0s. You may be thinking, okay, my computer only talks in 1s and 0s. How do I communicate with it? Think of it like this. We use the letters of the alphabet to form words and we give those words meaning. We use them to create sentences, paragraphs, and whole stories. The same thing applies to binary, except instead of A, B, C, and so on, we only have 0 and 1 to create words that we give meaning to. In computing terms, we group binary into 8 numbers, or bits. Technically, a bit is a binary digit. Historically, we used 8 bits because in the early days of computing, hardware utilized the base-2 numeral system to move bits around. 2 to the 8th numbers offered us a large enough range of values to do the computing we needed. Back then, any number of bits was used, but eventually the grouping of 8 bits became the industry standard that we use today. You should know that a group of 8 bits is referred to as a byte. So a byte of zeroes and ones could look like 10011011. Each byte can store one character, and we can have 256 possible values, thanks to the base-2 system, 2 to the 8th. In computer talk, this byte could mean something like the letter C. And this is how computer language was born. Let's make a quick table to translate something a computer might see into something we'd be able to recognize. What does the following translate to? Did you get hello? Pretty cool, right? By using binary, we can have unlimited communication with our computer. Everything you see on your computer right now, whether it's a video, an image, text or anything else, is nothing more than a 1 or a 0. It is important you understand how binary works. It is the basis for everything else we'll do in this course, so make sure you understand the concept before moving on.

1.3.2 Character encoding

Remember from the earlier video that a byte can store only zeros and ones. That means we can have 256 possible values. By the end of this video, you'll learn how we can represent the words, numbers, emojis and more we see on our screens, from only these 256 possible values. It's all thanks to character encoding. Character encoding is used to assign our binary values to characters so that we as humans can read them. We definitely wouldn't want to see all the text in our emails and Web pages rendered in complex sequences of zeros and ones. This is where character encodings come in handy. You can think of character encoding as a dictionary. It's a way for your computers to look up which human characters should be represented by a given binary value. The oldest character encoding standard used this ASCII. It represents the English alphabet, digits, and punctuation marks. The first character in ASCII to binary table, a lowercase a, maps to 0 1 1 0 0 0 0 1 in binary. This is done for all the characters you can find in the English alphabet as well as numbers and some special symbols. The great thing with ASCII was that we only needed to use 127 values out of our possible 256. It lasted for a very long time, but eventually it wasn't enough. Other character encoding standards recreated to represent different languages, different amounts of characters and more. Eventually they would require more than 256 values we were allowed to have. Then came UTF 8. The most prevalent encoding standard used today. Along with having the same ASCII table, it also lets us use a variable number of bytes. What do I mean by that? Think of any emoji. It's not possible to make emojis with a single byte, so as we can only store one character in a byte, instead UTF 8 allows us to store a character in more than one byte, which means endless emoji fun. UTF 8 is built off the Unicode Standard. We won't go into much of detail, but the Unicode Standard helps us represent character encoding in a consistent manner. Now that we've been able to represent letters, numbers, punctuation marks and even emojis, how do we represent color? Well, there are all kinds of color models. For now, let's stick to a basic one that's used in a lot of computers. RGB or red, green, and blue model. Just like the actual colors, if you mix a combination of any of these, you'll be able to get the full range of colors. In computerland, we use 3 characters for the RGB model. Each character represents a shade of the color and that then changes the color of the pixel you see on your screen. With just eight combinations of zeros and ones, were able to represent everything that you see on your computer, from a simple letter a, to the very video that you're watching right now on the Coursera website. Very cool. In the next video, we'll discuss how we actually generate the zeros and ones.

1.3.3 Binary

You might be wondering how our computers get these ones and zeros. It's a great question. Imagine we have a light bulb and a switch that turns the state of the light on or off. If we turn the light on, we can denote that state is one. If the light bulb is off, we can represent the state is zero. Now imagine eight light bulbs and switches, that represents eight bits with a state of zero or one. Let's backtrack to the punched cards that were used in Jacquard's loom. Remember that the loom used cards with holes in them. When the loom would reach a hole it would hooked to thread underneath, meaning that the loom was on. If there wasn't a hole, it would not hook the thread, so it was off. This is a foundational binary concept. By utilizing the two states of on or off, Jacquard was able to weave intricate patterns of the fabric with his looms. Then the industry started refining the punch cards a little more. If there was a hole, the computer would read one. If there wasn't a hole, it would read zero. Then, by just translating the combination of zeros and ones, our computer could calculate any possible amount of numbers. Binary in today's computer isn't done by reading holes. It uses electricity via transistors allowing electrical signals to pass through. There's an electric voltage, we would denote it as one. If there isn't, we would denote it by zero. For just having transistors isn't enough for our computer to be able to do complex tasks. Imagine if you had two light switches on opposite ends of a room, each controlling a light in the room. What if when you went to turn on the light with one switch, the other switch wouldn't turn off? That would be a very poorly designed loom. Both switches should either turn the light on or off depending on the state of the light. Fortunately, we have something known as logic gates. Logic gates allow our transistors to do more complex tasks, like decide where to send electrical signals depending on logical conditions. There are lots of different types of logic gates, but we won't discuss them in detail here. If you're curious about the role that transistors and logic gates play in modern circuitry, you can read more about it in the supplementary reading. Now we know how our computer gets its ones and zeros to calculate into meaningful instructions. Later in this course, we'll be able to talk about how we're able to turn human-readable instructions into zeros and ones that are computer understands through a compilers. That's one of the very basic building blocks of programming that's led to the creation of our favorite social media sites, video games, and just about everything else. And I'm super excited to teach you how to count in binary, that's up next.

1.3.4 How to count in binary

Binary is the fundamental communication block of computers, but it's used to represent more than just text and images. It's used in many aspects of computing like computer networking, which you'll learn about in a later course. It's important that you understand how computers count in binary. We've shown you simple lookup tables that you can use like the ASCII to binary table, but as an IT support specialist, whether you're working on networking or security, you'll need to know how binary works. So let's get started. You'll probably need a trusty pen and paper, a calculator, and some good old-fashioned brain power to help you in this video. The binary system is how our computers count using ones and zeros, but humans don't count like that. When you were a child, you may have counted using ten fingers on your hand. That innate counting system is called the decimal form or base-10 system. In the decimal system, there are 10 possible numbers you can use ranging from zero to nine. When we count binary, which only uses zero and one, we convert it to a system that we can understand, decimal. 330, 250, 2, 40, 4 million, they're all decimal numbers. We use the decimal system to help us figure out what bits our computer can use. We can represent any number in existence just by using bits. That's right. And we can represent this number just using ones and zeros. So how does that work? Let's consider these numbers: 128, 64, 32, 16, 8, 4, 2, and 1. What patterns do you see? Hopefully, you'll see that each number is a double of the previous number going right to left. What happens if you add them all up? You get 255. That's kind of weird. I thought we could have 256 values for a byte. Well, we do. The zero is counted as a value, so the maximum decimal number you can have is 255. What do you think the number is represented here? See where the ones and the zeros are represented. Remember, if our computer sees a one, then the value was on. If it sees a zero, then the value is off. If you add these numbers up, you'll get a decimal value. If you guessed 10, then you're right. Good job. If you didn't get it, that's okay too. Take another look. The 2 and 8 are on, and if we add them up, we get 10. Let's look at our ASCII to binary table again. The letter h in binary is 01101000. Now, let's look at an ASCII to decimal table. The letter h in decimal is 104. Now, let's try our conversion chart again. 64 plus 32 plus 8 equals 104. Look at that. The math checks out. Now, we're cooking. Wow! We've gone over all the essentials of the basic building blocks of computing and machine language. Next, you're going to learn how we build on top of this layer of computing to perform the task you'll do day to day.

1.4 Computer architecture layer

1.4.1 Abstraction

When we interact with our computers we use our mouse, keyboard or even a touch screen. We don't tell it the actual zeros and ones it needs to understand something. But wait, we actually do. We just don't ever have to worry about it. We use the concept of abstraction to take a relatively complex system and simplify it for our use. You use abstraction every day in the real world, and you may not even know it. If you've ever driven a car, you don't need to know how to operate the transmission or the engine directly. There's a steering wheel, some pedals, maybe a gear stick. If you buy a car from a different manufacturer, you operate it in pretty much the same way even though the stuff under the hood might be completely different. This is the essence of abstraction. Abstraction hides complexity by providing a common interface, the steering wheel, pedals, gear stick, and gauges in our car example. The same thing happens in our computer. We don't need to know how works underneath the hood. We have a mouse and a keyboard we can use to interact with it. Thanks to abstractions, the average computer user doesn't have to worry about the technical details. We'll use this under the hood e metaphor throughout the program to describe the area that contains the underlying implementation of the technology. In computing, we use abstraction to make a very complex problem, like how to make computers work, easier to think about. We do that by breaking it apart into simpler ideas that describe single concepts or individual jobs that need to be done, and then stack them in layers. This concept of abstraction will be used throughout this entire course. It's a fundamental concept in the computing world. One other simple example of abstraction in an IT role that you might see a lot is an error message. We don't have to dig through someone else's code and find a bug. This has been abstracted out for us already in the form of an error message. A simple error message like file not found actually tells us a lot of information and saves us time to figure out a solution. Can you imagine if instead of abstracting an error message our computer did nothing and we had no clue where to start looking for answers? Abstraction helps us in many ways that we don't even realize.

1.4.2 Computer architecture overview

In the last video I mentioned that people don't need to understand how a computer works for them to use it, because abstraction makes things simpler for us. That's technically true, but since you're stepping into the world of IT, you do need to understand all the layers of a computer and how they work. It's essential that you understand how the different pieces interact so you can resolve any issue that may arise. For the rest of this course we'll deep dive into the layers of computer architecture, and learn all the parts that make up a computer. A computer can be cut into four main layers, hardware, operating system, software, and users. The hardware layer is made up of the physical components of a computer. These are objects you can physically hold in your hand. Laptops, phones, monitors, keyboards, you get the idea. In the next lesson you'll learn all of the components of the computer and how they work. You'll even be able to build your own computer by the end of this module. The operating system allows hardware to communicate with the system. Hardware is created by many different manufacturers. The operating system allows them to be used with our system, regardless of where it came from. In the next few lessons, you'll learn about the major operating systems that we use today, and you'll be able to understand all of the underlying components that make up an operating system. By the end of these lessons, you'll have a strong grasp on the major components of any operating system, like Android or Windows, and use that knowledge to navigate any operating system. The software layer is how we as humans interact with our computers. When you use a computer, you're given a vast amount of software that you interact with, whether it's a mobile app, a web browser, a word processor, or the operating system itself. Later in this course, we'll learn how software is installed on our systems, and how we interact with different types of software. The last layer may not seem like it's part of the system, but it's an essential layer of the computer architecture, the user. The user interacts with the computer and she can do more than that. She can operate, maintain, and even program the computer. The user layer is one of the most important layers we'll learn about. When you step into the field of IT, you may have your hands full with the technical aspects, but the most important part of IT is the human element. While we work with computers every day, it is the user interaction that makes up most of our job, from responding to user emails to fixing their computers. By the end of the course, you'll also learn how to apply your knowledge of how a computer works to fix real world issues that can sometimes seem random and obscure. We'll do this by learning how to utilize problem solving tactics to identify issues and solutions. There's a lot ahead. The next instructor you're going to meet is a friend of mine, Devon. And I know there's no better person to teach you about hardware. He'll even show you how to build a computer from its component parts, pretty cool. But before you get to building that computer, we've got a quiz coming up for you on binary counting.

1.4.3 Kevin advice

We have a lot of people that have non-traditional backgrounds that have made here at Google. I've worked with people who have history degrees, I've worked with people who have economic degrees, and they're writing scripts, automating us, and how we can process these credits for this coins. I think people have a misconception that you have to have a traditional path in order to like succeed in IT. A lot of people have followed the traditional path, a lot of people do succeed following that traditional path, but I think the benefit of IT is that in the end, people just want to know whether or not you can fix the problem. Make sure you have strong fundamentals. They do and no coming back. A lot of time people think that, "Oh like I'm not going to need to worry about how to do. I don't even know, like understand the TCP IP model or the OSI model." That's like low level stuff. I can focus specifically on this one particular application or program that I'm going to be dealing with. There are instances where you'll run into problems where in having that foundational knowledge, will be very integral to solving the problem. As long as you're able to get to a point where you feel comfortable working with users, fixing their problems and supporting them, and the best way for you and them, you're going to always be viable in the world of IT.