Python Programming from Beginner to Paid Professional

Part 1

Learn Python for Automation & IT

with Video Tutorials

By

A. B. Lawal & A. J. Wright

Python Programming from Beginner to Paid Professional

ISBN: 9791220820387

Published in the United States

Table of Contents

About the Authors

A. B. Lawal and A. J. Wright have well over 15 years of software development experience. In the last few years, we used our experience to develop solutions using various programming languages on Windows, Linux, MacOS and PLCs (programmable logic controllers). We also built solutions from scratch and went as far as modifying open-source software to meet our client's needs.

We study hard and carry out researches constantly. We also love helping people get jobs or making their businesses successful. Nowadays, we work with a dedicated team of Python programmers who look into specific automation problems and proffer lasting solutions to them.

How this Book can Help You

This is not just *another* Python programming book. It is an intensive and *practical* Python programming course. It is part 1 of a 3-part series which serves as my exhaustive collection of step-by-step tutorials on the latest version 3 of Python programming language. It is a self-paced course that is excellent for beginners and accomplished experts alike. If you want to have fun learning or revising your Python programming with ease, this is the right course for you.

You will find this book indispensable if you are a computer programmer, an automation engineer or professional, a system administrator working in an IT firm, a data analyst/journalist, an educator, a computer science student or just anyone looking to acquire Python programming skills they need to succeed in their job or career. Yes, this course is exactly what you need to become a Pythoneer or Pythonista.

This course has 6 modules spread out over 25 chapters of both rich text and visual tutorials. You're not in this alone. I'm going to help you through it. Watching people coding is very different from learning how to code. So, you will not only be learning Python in this course, you will also be *doing.*

As you complete the tutorials, you're also going to get tested *a lot* on the materials we are covering by following Python best practices. Although this is a self-paced course, I strongly recommend that you complete it in not more than 6 weeks. For example, if you can complete one module every week, you can finish the course in 6 weeks.

To fully understand the basics of Python 3 programming, I strongly recommend you watch all the **53 in-depth HD videos** which are available in the **course resources folder** that you can download. The link for download is in Chapter 25 of this book. These video tutorials simplify everything you need to understand, and can help you **speed up your learning**.

Important terms and definitions discussed in this book are printed in bold texts, **like this**. Practice quizzes and answers are included at the end of each chapter to help you test how much you have improved. Go to Chapter 25 right now. You will find the link to the **course resources folder**. Once you open this link, you will be able to **download all the course videos, graded assessments and their solutions, projects and handy cheat sheets that give you all the information you need at a glance.**

How to Use the Video Tutorials, Programs & Practice Exercises

Except for revision purposes, the videos are not intended to be watched in isolation without first studying the content of this book. You should watch each video when you reach the point where it is referenced. This will ensure you fully understand the concept being discussed. The serial number and title (file name) of the relevant video to watch is stated at the point where it is referenced.

To have a solid foundation of Python, you must have an in-depth knowledge and develop Python coding skills quickly. Therefore, I strongly advise you attempt all the practice exercises like quizzes, graded assessments and projects in this book *on your own*, and *before* you check the answers and solutions provided. Write your own program as soon as you see me write mine in each video. Then cross check your programs with mine. Feel free to pause or rewatch the videos any time. You can also use or modify any of my codes as you wish.

Since I assume you have no knowledge of Python programming, I prepared this course in such a way that when you study it along with the accompanying videos (**53 episodes**), you will not only have an in-depth knowledge of Python 3 programming, you will also gain a lot of job experience you need to build automation and innovation, and earn higher salary.

This book begins with the fundamental knowledge you need to start writing your very first Python script. It goes on to teach the advanced topics you need to become a paid professional in the field of Python programming. So, after completing this course you will have a clear understanding of Object-oriented Programming and be able to apply it to real world industrial automation.

The methods presented in this book and the accompanying videos are those that are usually employed in the real world of IT and are the important ones you really need to learn. So, the information in this book is very valuable, not only to those who are just starting out, but also to other intermediate Python programmers.

Merely reading a copious Python programming book, or referring to Python help contents, is far from enough for learning how to “speak” Python language fluently. This book, unlike many others, takes the practical approach. It is designed in a course format with rich practice quizzes, assessments, challenge labs and interesting projects (with solutions) to engage you and give you hands-on practical experience.

First, this book will give you a big head start if you have never written a line of code before. Then it will teach you the techniques you need to learn, design and build anything from simple to complex and very useful Python programs.

Why Learn Python Programming?

If you learn Python, you will go mainstream. In case you haven't noticed, there are hundreds of today's most successful technology/IT companies using Python programs, such as Google, Netflix, Instagram, Reddit, Lyft, Spotify and so many others. Python is also being used at Bloomberg, the New York Times, and even at many local banks.

Python has many clear paths to finding meaningful job and work. For example, here is the link to apply to top remote USA Python developer Jobs.

Although some of these potential jobs are quite obvious - such as becoming a Python developer - there are other careers where the knowledge of Python is an asset that are more unexpected.

Module 1

"If milk gets bad it becomes yoghurt. Yoghurt is more valuable than milk. If it gets even worse, it turns to cheese. Cheese is more valuable than yoghurt. You are not bad because you made mistakes. Mistakes are the experiences that make you more valuable as a person. Christopher Columbus made a navigational error that made him discover America. Alexander Fleming's mistake led him to invent Penicillin. Don't let your mistakes get you down. It is not practice that makes perfect. Our mistakes enable us to learn to make ourselves perfect!" - Josh Aisosa

1. The Foundation: Getting Started with Python Programming

In this module I'll introduce you to the book's course format. Then, we'll dive into the basics of programming languages and syntax, as well as automation using scripting. I'll also introduce you to the Python programming language and some of the benefits it offers. Last up, we'll cover some basic functions and keywords of the language, along with some arithmetic operations.

1.1. Specialization Introduction

Working in IT is more than just a job. It's a career path. Research shows that the field of IT support is a launchpad for future career growth and better wages. In fact, a study on the subject was recently conducted by the Harvard Business School Accenture and Burning Glass entitled "Bridge the Gap".

It found that among today's middle-skilled jobs which require training but not a formal college degree, IT support offers clear pathways to prosperity. I saw this phenomenon play out at Google, in their IT support program. Those who push themselves to learn how to code in Python typically saw strong career growth.

They built skills that are critical to accessing higher level positions in the IT field, and after honing those skills through hard work and determination, they advanced into more technical IT support specialists. Some of them are systems administrators, technical solutions engineers, and even site reliability engineers. The common thread across all of these roles is knowing how to write code to solve problems and automate solutions.

By expanding your toolbox to include coding skills, you open a window into the world of systems management that can lead you towards more advanced technical roles down the line. Python in particular is having a huge surgeon's. According to the 2019 Stack Overflow Developer Survey, Python is the coding language most people want to learn. The second most loved by those who already know it and the fourth most popular overall.

So why take this course to learn how to code in Python? Well, first it's geared towards people

who are already in or aspiring to be in the field of IT. Maybe you're thinking bigger about your current IT role and want to work towards managing operations at scale, or maybe you're just starting out and looking to break into the IT industry.

Perhaps you've taken an IT Support Professional Certificate program somewhere already, or you have equivalent IT support knowledge with basic computing skills, like working with files and directories, familiarity with networking concepts, and understanding how to install software on your computer. In any case, this course is tailor-made for you. Second, this course offers three hands-on methods of teaching coding, Python and automation: Code blocks, Jupyter notebooks, and labs.

Third, I worked with an awesome group of Googlers who helped me prepare this course. They all started their careers in IT support, then learned programming and moved onto more technical roles like me. I can't wait to share our stories with you on how we use Python in our day-to-day activities. So, I will introduce you to the Python programming language with a special focus on how this language applies to automating tasks in the world of IT systems support and administration. I'm super excited to teach you this course.

When I was younger, I had no idea that careers in IT even existed. Later, I remember going to a System Administration Summit where there were hundreds of men and about three women that were sysadmins. A lot has changed since then but there's still so much we can do to bring new ideas and representation into the IT field. That's why I want to share my knowledge with as many people as possible (and the reason I prepared this book in a rich course format).

I love my Python programming and I love the people I work with because they make it easy to ask for help and offer their guidance. This type of support network allows our team and ultimately our industry to be more successful. I understand from experience that it can feel pretty intimidating and maybe even a bit scary to learn a coding language.

Just remember, everyone started where you are right now with the first command, the first script, and of course, the first of many errors. When I started out in my career, I strive to get everything perfectly right the first time I tried it. But that actually slowed down my progress. So, don't be afraid to make mistakes, it will give you a leg up. So, let's get down to it. What's ahead?

This book (Part 1) begins with a crash course in Python where you will learn to write simple programs and understand their role in automation. Next (Part 2), we'll get more hands-on focus on how Python interacts with the operating system. After that, we'll cover how to use Git and GitHub to manage versions of your code. Then we will focus on troubleshooting and debugging techniques to find and solve the root cause of problems in IT infrastructure.

The next course (Part 3) covers automating at scale where you will learn to deploy configuration management on a fleet of either physical or virtual machines running in the Cloud.

Last up, we will bring all this knowledge together and complete two final projects designed to solve tasks that you might encounter in real-world IT settings. You can post your projects to GitHub to show off your fancy new skills to employers or friends or both.

That was a lot to rattle off! Are you excited? You're in very good hands!

So, let's get ready to learn some new skills and maybe even have some laughs along the way. Let’s dive in straight to the next section.

1.2. Course Introduction

If you work in IT, computer programming skills open up an incredible amount of opportunity. Being able to write scripts and programs that tell your computer to perform a task equips you with an invaluable tool. Not only does it make your work easier and more efficient, it can help you grow faster and advance further in your IT career.

But how do you even start to learn a programming language like Python? How do you recognize when to tell a computer to perform a task? And how do you then write a program to actually get your computer complete the task you want it to do? The thought of learning to write a program in Python can make you feel a whole bunch of emotions excitement, anticipation that feeling of wanting to dive right in and get going and also fear.

You might ask yourself, can I really learn how to code or do I have it in me? I'm here to tell you, yes, you can absolutely do this. Learning how to program can be scary and intimidating, but at the same time it's really fun and really exciting. In coding, as in as in life if we're going to get philosophical, the most rewarding work is usually a bit challenging, but ultimately well worth the effort. Of course, I'm able to say all this from experience, especially the cheesy parts.

The role of a sysadmin can vary a lot from company to company and even within different teams in the same company. I happen to work in the corporate identity and access management operations team, which is a really long way of saying that we make sure that everyone is represented correctly and if they need to access certain resources, they can.

What I love the most about being a sysadmin is that the role has so many diverse functions. We handle loads of unique problems and edge cases from tinkering with different systems to collaborating with other teams. I am always learning something new, so it's really hard to get bored.

It all starts with knowing how to automate, if you're an IT support specialist, a systems administrator, or in a role somewhere in between knowing how to get computers to do the hard work for you will set you apart from others in similar IT roles and make your life much easier. Think about it, would you rather manually deploy 100 computers on your own or tell your computer to do it all for you all at once?

No-brainer, right? Having a coding skill set can help you grow into more specialized roles like a systems administrator, Cloud Solutions engineer, DevOps specialist, site reliability engineer, or who knows, maybe even web developer or data analyst. The point is, being able to write a program is an essential tool in your IT toolkit and more and more employers are looking for these skills in the people they hire.

If you've ever learned a new skill, like playing a musical instrument, speaking a foreign language, knitting, or skateboarding, you know that getting good at something new requires a lot of practice. For me, I love to learn new languages and I'm proud to say I now speak Spanish, Arabic, French, and I even know ten words in Russian.

Our world is shaped by the words and the languages we speak and while some words may be unique to one language you can always find similarities that help you learn and understand. Being able to connect the dots between cultures allows me to see things others might not, kind of sounds like this applies to IT programming, huh? My point is, whether you're learning French or Python, it 's never easy.

You have to start small, learn the basics and practice those until you master them. Only then can you move on to more complex and impressive stuff. We'll start slow, master the foundation's together and you'll soon be ready for more challenging stuff. By the end of this course, you'll understand the benefits of programming in IT roles.

You'll be able to write simple programs using Python, figure out how the building blocks of programming fit together, and combine all this knowledge to solve a complex programming problem. That's right, by the end of this course, you're going to write a program in Python that's designed to solve a real-world IT problem. Super exciting right?

We'll start off by diving into the basics of writing a computer program. You'll get hands-on experience with programming concepts through interactive exercises and real-world examples. You'll quickly start to see how computers can perform a multitude of tasks. You just have to write code that tells them what to do. Along the way, we'll be talking about automation, which is process of getting computers to automatically do a task that us humans normally have to do by hand.

Now, some of the stuff can get a little complex and confusing. I promise to do my best to make

these lessons clear and easy to understand, but if you get stuck at any point, please feel free to re-watch the videos. Practice as much as you like and take the time you really need to understand these topics. The goal of this course isn't to teach you everything there is to know about software engineering because yikes, that would be a long course.

Instead, I'm going to introduce you to some of the key concepts of programming and scripting that will empower you to spot opportunities for automation in real life. You're about to learn a skill that can help you take your career to whole new levels. Are you excited? I'm excited too, so let's jump in!

1.3. A Minute to Set Yourself up for Success

Did you know that people tend to get more out of a course like this, and are more likely to succeed when they set their intention to complete the course before they start? Use a pen and paper to complete your commitment statement below and help yourself reach your goals. Don't worry – this is just for you and is not graded.

***

I commit to my goal to complete this course. I choose to complete it because…

(*Finish the sentence. What motivated you to take this course?*)

When I run into obstacles or lack motivation at some point during the course, I will …

(*Finish the sentence. What would you do or say to motivate your future self?*)

(Finally, *commit by writing your name here)*

***

Make sure you save your commitment where you can see it everyday.

1.4. Welcome to the Course

This course is designed to teach you the foundations of programming in Python. We're excited to join you on this journey as you learn one of the most-in-demand job skills in IT today. In the U.S. alone, according to Burning Glass data from May 2019, there were approximately 530K job openings in 2018 asking for Python skills. This course requires no previous knowledge of programming.

1.4.1. How the 6-Week Deadline Works

I deliberately set a deadline of one week for each module for you to complete this course.

Heads up: These deadlines are there to help you organize your time, but you can take the course at your own pace. If you "miss" a deadline, you can just reset it to a new date. There's no time limit in which you have to finish the course, and you move on to the second part of the course whenever you finish.

1.4.2. Getting and Giving Help

Here are a few ways you can give and get help:

1. Official Python Discussion forums: You can share information and ideas with your fellow learners in the Python discussion forums. These are also great places to find answers to questions you may have. If you're stuck on a concept, are struggling to solve a practice exercise, or you just want more information on a subject, the discussion forums are there to help you move forward.

2. Contact Us: Use the Contact us link (email link) at the end of Chapter 25 to request information on specific issues. These may include error messages, challenge labs, projects, and problems with video download or playback.

1.4.3. Finding Out More Information

Throughout this course, I'll be teaching you the basics of programming and automation. I'll provide a lot of information through videos and text readings. But sometimes, you may also need to look things up on your own, now and throughout your career. Things change fast in IT, so it's critical to do your own research so you stay up to date on what's new. We recommend you use your favorite search engine to find additional information— it's great practice for the real world!

On top of search results, here are some good programming resources available online:

The official Python tutorial. This tutorial is designed to help people teach themselves Python. While it goes in a different order than the one I am taking here, it covers a lot of the same subjects that we explore in this course. You can refer to this resource for extra information on these subjects.

The official language reference. This is a technical reference of all Python language components. At first, this resource might be a little too complex, but as you learn how Python works and how it's built, this can be a useful reference to understand the details of these interactions.

1.5. Official Python Discussion Forums: Join, Meet & Greet

Join, meet and greet your learner community! You're now a part of a growing group of learners

all investing in their future by learning how to program in Python.

Say hi! Write a brief introduction to help other learners get to know you better. Not sure what to write? Here's a few tips:

- **Hopes and goals**. Why are you taking this course? What are your expectations? What excites you most about learning to program with Python? What do you hope to achieve once you've completed this course?
- **Hobbies and interests**. What other things are you interested in, besides programming or IT? You might find fellow learners with the same interests!
- **Where are you from.** There are learners from all over the globe! Why not share with others which part of the world you live in? But remember, don't share specific details like your personal mailing address.

You don't have to take part. But it's more fun if you do!

2. Introduction to Programming

2.1. The Beginning of Your Programming Journey

As the Chinese proverb says, a journey of a thousand miles begins with a single step. Today's a big day, you're taking your first step in your journey to learn how to write scripts in Python. It's going to be a little challenging at times, but really, it's not that scary. We'll go slow and give you everything you need to fully grasp each concept before we move along.

In the next few sections, you'll discover the fundamental concepts of computer programming. You'll learn what a programming language is, what scripting is, what languages are out there other than Python, and how this all relates to IT. We'll also have you coding before you know it with small coding exercises we've cooked up to give you hands-on practice with Python. This will include writing your very first Python script. But always keep in mind, if at any point along the way you feel lost or confused, don't panic.

You can read any section again or watch the videos as many times as you need to let the concept sink in, plus you can ask questions in the discussion forums, which is one of the best ways to find extra information and connect with other learners. When I was asked to participate in this program, it made me think about when I first started to code. If I could give that younger version of myself a piece of advice, this is what I would tell her: it never works the first time.

Seriously, as a newbie, I expected it all to work like magic. I thought that following the rules and getting it right the first time would prove my value as a coder, but that's just not true, not even the best of the best. If you expect to write perfect code on the first shot, you're going to be disappointed. You hear that younger self? Try not to feel overwhelmed by the details. Connecting the dots only comes with experience, so the best way to learn is to just jump in.

The truth is everyone learns at their own pace. If you already know some of these concepts, feel free to skip ahead to the part that interests you the most. If you're starting from scratch, take as long as you need for each concept. The assessments will be right there waiting for you when you're done, and if at any point you start doubting yourself, remember, even the most advanced programmers started thinking, Python… what's Python?

Well, we're about to learn all about it, so let's dive in to do a rundown of what programming is.

2.2. What is Programming?

At a basic level, a computer program is a recipe of instructions that tells your computer what to do. When you write a program, you create a step by step recipe of what needs to be done to

complete a task and when your computer executes the program it reads what you wrote and follows your instructions to the letter. How nice is that? The recipe is written in a code called *programming language*.

Programming languages are actually similar to humans spoken languages since they have a *syntax* and *semantics*. Now if it's been awhile since your last grammar class, here's a quick refresher on syntax and semantics.

In a human language, **syntax is the rules for how a sentence is constructed** while **semantics refers to the actual meaning of the statements**. In English, sentences generally have both a subject, that's a person, place, or thing, and a predicate usually a verb and a statement that explains what the subject is doing.

Let's take the sentence, *Paula loves to program in Python* as an example. In this sentence, Paula is the subject and loves to program in Python is the predicate. To form a sentence that others can understand, you need to know both the syntax that constructs the sentence and the semantics that gives it meaning. The same applies to programming languages.

In a programming language like Python, the syntax is the rules for how each instruction is written and the semantics is the effects the instructions have. Much like spoken languages, there are lots of programming languages to choose from. Each has its own history, features, and applications but they all share the same fundamental ideas. So, once you understand the basic concepts in one programming language, it becomes much easier to learn another.

Lastly, computers always do exactly what they're told. So when you write a program, it's important to be super clear about what you want the computer to do. Learning the syntax and semantics of the programming language you choose will allow you to do just that. Make sense? Before we continue, let's spend a moment on terminology.

2.2.1. Difference Between *Script* and *Program*

In the next few sections you'll hear the term script being used a bunch. So, what's the difference between a *script* and a *program*? The line between the two can be a bit blurry. In this course, we'll use the terms interchangeably. In general, you can think of scripts as programs with a short development cycle that can be created and deployed rapidly.

In other words, a script is a program that is short, simple, and can be written very quickly. In this course we'll focus on a specific scripting language called python which we'll use to learn the basics of programming. We'll learn about the python syntax, the rules of how to write a python program, and the semantics or meaning of the different pieces involved.

Before we start learning how to code and having you write your first python script, let's talk

more about what automation is and why it's useful.

2.3. What is Automation?

Although we might not realize it, we reap the benefits of automation all the time in our daily lives. Do you ever pay your bills with scheduled payments or use a self check out at the grocery store? I always set my coffee machine to start brewing before I've even gotten out of bed. The promise of fresh coffee makes early mornings way easier.

Automation is the process of replacing a manual step with one that happens automatically. Take a traffic light for example, which continuously regulates the flow of vehicles at an intersection. A traffic light requires a human intervention only when it needs repairs or maintenance.

The automatic regulation of traffic means that humans don't have to stand at the intersection manually signaling when cars should stop or go. Instead, people can concentrate on more complex, creative, or difficult tasks like focusing on where you're driving. What's more, traffic lights don't get tired, bored, or accidentally display a green light when they meant red. This highlights another benefit of automation consistency.

Let's face it, us humans are flawed and sometimes we make mistakes, a human performing the same tasks hundreds of times will never be as consistent as a machine doing the same thing. But for all its advantages, automation isn't a solution for every situation, some tasks just aren't suited for automation.

For example, they may require a degree of creativity or flexibility that automatic systems can't provide or for more complicated or less frequently executed tasks creating the automation may actually be more effort or cost than it's worth. Think about when you get a haircut. What would it take to automate the actions of cutting hair with a machine?

The client’s height, the shape of their head, their current hair length, and desired hairstyle would all need to be taken into account when designing the automatic system. We need to replicate the creativity and skills of a trained specialist along with extensive testing to ensure the clients safety and quality haircut. And if you've ever had a bad experience at a hair salon, you know quality can be subjective.

In this case, the cost and effort of automation just isn't worth the benefits of an automatic haircut would provide which is why we don't have robot hairstylists. Not too complex, right? Automation is a powerful tool when used in the right place at the right moment. It can save time, reduce errors, increase consistency, and provide a way to centralized solutions and mistakes making them easier to fix.

Throughout this course, and upcoming ones we'll be talking about when it makes sense to apply automation and exactly how you do it. Eventually knowing when and where to use automation will become automatic for you.

2.4. Getting Computers to Work for You

Working in IT, a lot of what we do boils down to using a computer to perform a certain task. In your job you might create user accounts, configure the network, install software, backup existing data, or execute a whole range of other computer-based tasks from day to day.

Back in my first IT job, I realized that every day I sat at my computer to work I typed the same three commands to authenticate into systems. Those credentials timed out everyday by design, for security reasons, so I created a script that would automatically run these commands for me every morning to avoid having to type them myself.

Funny enough, the team that monitors anomalous activity discovered my little invention and contacted me to remove it, oops! Tasks performed by a computer that need to be done multiple times with little variation are really well suited for automation, because when you automate a task you avoid the possibility of human errors and reduce the time it takes to do it.

Imagine this scenario, your company had a booth at a recent conference and has gathered a huge list of emails from people interested in learning more about your products. You want to send these people your monthly email newsletter, but some of the people on the list are already subscribed to receive it.

So how do you make sure everyone receives your newsletter, without accidentally sending it to the same person twice? Well, you could manually check each email address one by one to make sure you only add new ones to the list, sounds boring and inefficient, right?

It could be, and it's also more error prone, you might accidentally miss new emails, or add emails that were already there, or it might get so boring you fall asleep at your desk. Even your automated coffee machine won't help you out there. So, what could you do instead?

You could get the computer to do the work for you. You could write a program that checks for duplicates, and then adds each new email to the list. Your computer will do exactly as it is told no matter how many emails there are in the list, so it won't get tired or make any mistakes.

Even better, once you've written the program you can use the same code in the future situations, saving you even more time, pretty cool, right? It gets better. Think about when you're going to send these emails out. If you send them out manually, you'll have to send the same email to everybody, personalizing the emails would be way too much manual work.

If instead you use automation to send them, you could have the name and company of each person added to the email automatically. The result? More effective emails, without you spending hours inserting names into the text. Automating tasks allows you to focus on projects that are a better use of your time, letting computers do the boring stuff for you.

Learning how to program is the first step to being able to do this. If you want to get computers to do the work for you, you're in the right place. Earlier in this section, I told you about the first task I ever automated, now I want to tell you about the coolest thing I ever automated.

It was a script that changed a bunch of access permissions for a whole lot of my company's Internal Services. The script reversed a large directory tree with tons of different files, checked the file contents, and then updated the permissions to the services based on the conditions that I laid out in the script.

Okay, I admit I'm a total nerd, but I still think it's really cool. Next up, it's time to share your ideas. What things would you like to automate using programming? While these forum discussions are optional, they're really fun. Seriously, they let you get to know your fellow learners a bit and collaborate on ideas and insights.

Make sure you read what others are saying, they may give you ideas that you haven't even thought of. After that, you're ready to take your very first quiz of the course. Don't worry, it's just for practice.

2.5. Discussion Forums: Your Hopes for Automation

What everyday tasks would you like to automate through programming? Share your ideas, hopes, and goals with your fellow learners.

2.6. Practice Quiz 1: Introduction to Coding in General - 5 Questions

Open a fresh notepad or Word file in your computer to answer this quiz.

Time Allowed: **15 minutes**.

1. What's a computer program? Select only the correct response. – *1 point*

A. A set of languages available in the computer.
B. A process for getting duplicate values removed from a list.
C. A list of instructions that the computer has to follow to reach a goal.
D. A file that gets copied to all machines in the network.

2. What's the syntax of a language? Select only the correct response. – *1 point*

A. The rules of how to express things in that language.
B. The subject of a sentence.
C. The difference between one language and another.
D. The meaning of the words.

3. What's the difference between a program and a script? Select only the correct response. – *1 point*

A. There's not much difference, but scripts are usually simpler and shorter.
B. Scripts are only written in Python.
C. Scripts can only be used for simple tasks.
D. Programs are written by software engineers; scripts are written by system administrators.

4. Which of these scenarios are good candidates for automation? Select all that apply. – *1 point*

A. Generating a sales report, split by region and product type.
B. Creating your own startup company.
C. Helping a user who's having network troubles.
D. Copying a file to all computers in a company.
E. Interviewing a candidate for a job.
F. Sending personalized emails to subscribers of your website.
G. Investigating the root cause of a machine failing to boot.

5. What are semantics when applied to programming code and pseudocode? – *1 point*

A. The rules for how a programming instruction is written.
B. The difference in number values in one instance of a script compared to another.

C. The effect the programming instructions have.
D. The end result of a programming instruction.

The correct answers are provided below. Use them to grade your performance (X out of 5) in this quiz. Always use the correct answers provided to grade your performance in each of the subsequent practice exercises in this book. **X** is the total number of correct answers you get out of a total of 5. Don't forget to save your result!