Chapter 13 Computer Languages, Applications, and Emerging Technologies

Shambhavi Roy

Clinton Daniel
University of South Florida

Manish Agrawal
University of South Florida

Follow this and additional works at: https://digitalcommons.usf.edu/dit_tb_eng

Scholar Commons Citation
https://digitalcommons.usf.edu/dit_tb_eng/12

This Book Chapter is brought to you for free and open access by the The Modernization of Digital Information Technology at Digital Commons @ University of South Florida. It has been accepted for inclusion in FUNDAMENTALS OF INFORMATION TECHNOLOGY: Textbook – English by an authorized administrator of Digital Commons @ University of South Florida. For more information, please contact digitalcommons@usf.edu.
# Chapter 13—Computer Languages, Applications, and Emerging Technologies

## Overview

**Types of Computer Languages** 278

- Binary Code 279

**Computer Programming** 279

- Block Based Coding 280

## Programming Basics

- Variables 281
- Functions/Methods 282
- Algorithm 284
- Object Oriented Programming 285
  - Objects 286
  - Popular OOP Languages 286

## Applications

- Desktop Applications 288
- Web Applications 288
- Mobile Applications 289
- The Interplay Between Desktop, Web, and Mobile 289
- Computer Program Design 289

## Emerging Technologies

- Artificial Intelligence (AI) and Machine Learning (ML) 290
- Blockchain and Cryptocurrencies 292
- Augmented Reality and Virtual Reality 292
  - Internet of Things 293
- Autonomous Vehicles (Self-Driving Cars, EVTOL, Drone Delivery) 293
- Advanced Robotics 294
- Biotechnology 294

## Learning to Program: Getting Started

Chapter Terms and Definitions 298

Chapter Case: Vivian’s Raspberry Pi 299
“Computer language design is just like a stroll in the park. Jurassic Park, that is.”
—Larry Wall, creator of the Perl programming language

Overview

Computer languages are the rules for writing computer programs. Just as we need languages such as English to communicate ideas and information with each other, we need computer languages to tell computers what to do. Computer languages are popularly called programming languages. The important difference between human languages and computer languages originates from the human ability to handle ambiguity and fill context. For example, in most contexts people would understand the phrase “let’s meet tomorrow.” But computers would not be able to figure out who is meeting, the place of the meeting, the time of the meeting, any prior preparation for the meeting, etc. Therefore, computer languages use specific syntax and grammar to precisely communicate with machines to avoid miscommunication. Once you develop some familiarity with computer languages and comfort with giving precise instructions in computer languages to get the job done, you can create increasingly complex computer applications that make life easier. Computer applications are a set of instructions written in a programming language. Computers read these instructions and perform the corresponding set of actions.

Types of Computer Languages

Computer languages (programming languages) can be divided into a few broad categories for easier understanding: procedural programming languages, object-oriented programming languages, scripting languages, markup languages, domain-specific languages, and low-level languages. Here’s an overview of these categories:

- **Procedural programming languages** are computer languages that use precise steps to compose programs. In a way, all programming languages are procedural languages, but the term typically refers to languages with a limited set of data types such as numbers and strings. C, Fortran, and Pascal are examples of procedural languages and allow programmers to create procedures or subroutines to perform specific tasks. Today, procedural languages are primarily used for introductory programming classes. Historically, they were used to write the earliest scientific and engineering applications.

- **Object-oriented programming languages** are computer languages that allow developers to create their own data types by organizing data and related functions into objects. Object-oriented (OO) languages greatly simplify representing the real world in computer programs and are widely used in software development. Examples of object-oriented languages include Java, C#, and C++.

- **Scripting languages** are computer languages used to automate tasks using the capabilities of existing applications. Scripting languages are typically aimed at end users and are considered easier to learn than procedural or object-oriented languages. AppleScript is an example

---

of a scripting language for MacOS, and AutoHotKey\textsuperscript{205} is an example of a scripting language for Windows. JavaScript and Python are scripting languages that have evolved into powerful languages to create computer applications. Scripting languages are typically not used to create software for commercial distribution, since scripts are not compiled, and the programs can be read by all users.\textsuperscript{206}

- **Markup languages** are computer languages used to specify how information should be displayed or interpreted. HTML, Markdown and XML are well-known markup languages. Markup languages define markup tags, which are used to create webpages and other content that can be displayed on a variety of devices.

- **Domain-specific languages** are computer languages optimized for specific application domains. SQL is an example of a domain-specific example. Other domain-specific languages include R for statistical applications and MATLAB for engineering applications. Domain-specific languages greatly simplify application development for complex domains such as data retrieval (SQL) and statistical data analysis (R).

- **Low-level languages** are programming languages that are close to the processor’s native instruction set. They are sometimes called assembly language. Programs in all other languages (e.g., procedural, object-oriented, and domain-specific languages) are converted by compilers into low-level language programs for each type of processor.

### Binary Code

Eventually, all computer programs are stored as instructions in binary code. Computers can only read binary code, which is a collection of 1s and 0s. Binary code is the native language of computers and is necessary for communication and storage of data. For example, files and data are stored in binary format on hard drives and other storage media.

Specialized software programs called compilers convert software code, written in any of the above languages, into binary code that can then be executed directly by the computers. Compilers allow developers to write computer programs in languages that resemble plain English (called high-level languages) and convert these programs into binary code customized for each processor.

### Computer Programming

Each type of programming language has its own strengths and weaknesses. The choice of language often depends on the specific needs of the application being developed. It is common to use a combination of languages to build a software application.\textsuperscript{207} You may use HTML and JavaScript to


\textsuperscript{207} A good thread with examples of languages used in different popular applications is at https://www.quora.com/In-what-programming-language-s-is-Gmail-implemented (accessed June 2023).
manage the look and feel (frontend) of the application, and Java or C++ to create the middleware that handles the business logic and responds to requests from the frontend. You may use SQL to create your backend code that helps your middleware interact with the database to store and retrieve information.

Programming languages and frameworks are evolving rapidly to handle emerging business needs. These days, it is becoming increasingly common to use JavaScript to build the frontend as well as backend, so you just need to learn one language to build entire applications. This greatly improves developer productivity. Frameworks like React Native let you use JavaScript to build mobile applications. These frameworks also do the heavy lifting to convert your JavaScript code into the required low-level language components necessary to work with Apple IOS and Google Android phone systems.

We begin our introduction of programming languages with block-based programming. Block-based programming is a way to use graphical interfaces to write simple programs. If you have never tried computer programming before or if programming languages appear complicated, you could try block-based coding till you get comfortable enough to use regular programming languages.

If you would like to learn programming, we have some recommended resources at the end of the chapter.

**Block Based Coding**

Block coding is a visual programming language that uses blocks or graphical elements to represent programming concepts instead of traditional text-based coding. These blocks can be dragged and dropped to create a sequence of commands or instructions.

![Figure 227](image-url) — The drag and drop capability makes Blockly a great starting point when learning to code.
Blockly\textsuperscript{208} from Google is an example of a block-based computer language. Scratch from MIT labs is another block-based computer language that allows developers to create animations and stories. As shown in Figure 227, Blockly allows you to use simple graphical interfaces to specify instructions, and it converts these instructions into well-formed programs in different languages.

Block coding is a fun and interactive way to learn programming, ideal for beginners and even children. It allows you to focus on the logical structures of programming, without worrying about the syntax and details of text-based coding.

Block coding can give you a taste of the power of programming. It can also help you build foundational skills to assist in the move to text-based programming.

**Programming Basics**

While there are many popular programming languages (e.g., Java, C#, C, C++, JavaScript, Python), they all share most of the underlying concepts. Once you learn the basic programming concepts and use them in a few languages, learning new programming languages will be easy and fun. Here are a few concepts you will need to learn no matter which language you choose. If you would like to practice the examples in this chapter, and create your own programs, you can use the jdoodle online code editor. Most students find Python to be their favorite introductory language. The Python editor is at https://www.jdoodle.com/python3-programming-online/.

**Variables**

A variable is a named storage location in a computer’s memory that holds a value. Variables are the basic mechanism used to store and manipulate data in code. A variable is one of the first things you will learn when you begin to write software programs.

Let’s say you are creating a program that calculates the area of rectangles. Since the area is computed from the length and width of the rectangle, you would need to store the width and height of the rectangle as variables. You would need to create one variable for each dimension, maybe one called width and the other called height. For simplicity in this example, let’s assume all numbers are integers. Every language has its own way of declaring a variable. Once you declare the variables to hold the dimensions of the rectangle, you would assign values to the variables when a user inputs the width and height values of the rectangle.

Variable declaration of the type int (integer):

\begin{verbatim}
int width;
int height;
\end{verbatim}

Variable assignment:

\begin{verbatim}
width = 10;
height = 5;
\end{verbatim}

In programming, the equals operator (\(=\)) is typically used to assign values to variables. Once we have the values assigned to variables, we can perform calculations to get the area of the rectangle. If we wish to save this value for future use, we will need a third variable (Area) to store the value of the area:

```c
int area;
```

We can now compute the area as the product of the width and height as:

```c
area = width * height;
```

In the above statement, we ask the computer to fetch the values stored in width and height and multiply the two. The final output or the area is stored in area. As you see in the example above, computer programs written in modern programming languages read much like the same commands written in plain English.

To be useful to end users, just doing the calculations is often not enough. Users likely also want to see the results. You can print the output to the display using the print function available in most programming languages:

```c
print ("The area of the rectangle is:", area);
```

When users run your program and enter the height and the width values, they will see the following message:

```
The area of the rectangle is: 50.
```

**Functions/Methods**

A function (aka method in some programming languages) is a block of code that performs a specific task. A function is defined with a name and can be called or invoked repeatedly from other parts of a program.

Functions provide a way to modularize code and make it easier to read, understand, and reuse. Instead of writing the same code multiple times in different parts of a program, a function can be defined once and called whenever it is needed. Functions also improve program correctness since program errors only need to be fixed in one place (the method), instead of all the places where the methods are used.

Functions typically have inputs and outputs. The inputs are called parameters or arguments. They represent the data that the function will receive and work on. The outputs are the result of the function's computation and can be returned to the calling code.

Here is an example function in JavaScript that adds two numbers together and returns the result:

```javascript
function AddNumbers (x, y) {
    let sum = x + y;
    return sum;
}
```
In this example, the function `AddNumbers` takes two parameters `x` and `y`. The function adds `x` and `y` together and stores the result in a variable named `sum`. Finally, the function returns the value of `sum` to the calling code.

You can call this function from other programs by passing two numbers. Here’s how you could call the function `AddNumbers`:

```javascript
let result = AddNumbers(5, 7);
```

There are several ways to view the result. Developers typically print the values to the console to test their code. JavaScript has the “console.log” function to print results to the console. We can use the `console.log` function to print the value of the result variable as shown below (// is used to add a comment. Comments are meant for developer reference and are ignored by the computer processor. // 12 in the line below is a comment, indicating that the output is expected to be 12.):

```javascript
console.log(result); //Output: 12
```

**Figure 228** shows the complete example. On the left is the html page with the JavaScript method defined in lines 6–9. Clicking the “Add” button on the page calls the `AddNumbers` method and passes the values 5 and 7 as arguments to the method. The result is saved as the value of the variable, “result.” Finally, the result is printed to the console. To view the console, we open the Inspector (right-click anywhere in the browser and select “Inspect,” and go to the console tab. We see the result in the console (bottom half on the right). You can type this example in a file, save it with an html extension, open the file in a browser, and use the Inspector to observe how the program is executed. A convenient way to do that is to set breakpoints in the JavaScript method. The browser will stop execution when it reaches the breakpoint and show you the value of every variable. You can then step through the function line by line, observing how the variables change values as the code is edited.

![Figure 228 — An example of the AddNumbers method.](image)

Functions are an important part of programming languages and are used extensively in both frontend and backend development. They allow us to write reusable code that can be called from anywhere in our program, making our code more modular and easier to maintain.
Algorithm

An algorithm is a set of instructions for solving a problem. While any computer program that does the job can be considered an algorithm, in computer science and computer programming, algorithms typically refer to the most efficient methods to perform a task. There are well-known algorithms to efficiently perform specific operations such as sorting and searching for data. Advances in algorithms are one of the most important ways that Digital Information Technologies have improved lives across the globe.

Your algorithm could be just an idea shown as a flowchart or it could be pseudocode—a detailed set of instructions to write a program in the language of choice. Algorithms are language independent and programmers can implement them in any language they choose.

For example, here's an algorithm for a student to sign up for a class at the University of South Florida:

- begin by logging into the online registration system for the school;
- browse the course catalog to find the desired class;
- check for any prerequisites or restrictions on the class;
- select the desired class and verify the course details, including course number, schedule, location, and instructor;
- add the class to the student's schedule;
- check for any conflicts with the student's existing schedule or other classes;
- if there are no conflicts, confirm the enrollment in the class, else remove the class from the schedule;
- pay any associated fees or tuition for the class;
- confirm the enrollment and payment and review any important deadlines or course requirements;
- if there are conflicts or other issues preventing enrollment, contact the school's registration office for assistance;
- end the enrollment process.

Here are a few popular algorithms you may want to learn about or implement if you are interested in computer science.

- **Sorting algorithms** are used to sort collections of data. Sorted data is much easier to search than unsorted data, therefore, sorting is an essential task for any data-intensive application. Depending on the type of data you have, you may choose from one of the many available sorting algorithms: bubble sort, quick sort, merge sort, heap sort, and others.²⁰⁹

- **Search algorithms** are used to find a specific piece of information within a collection of data. Once again, depending on the type of data and other constraints, you could try out the various

search algorithms—linear search, binary search, jump search, exponential search, Fibonacci search, and others—to find the one that works best.\(^\text{210}\)

- **Encryption algorithms** are used to encrypt data to protect it from unauthorized access. AES, RSA, and DES are all examples of popular encryption algorithms.\(^\text{211}\)

- **Machine learning algorithms** have gained in importance and may change the way we analyze data or make predictions and decisions based on data. Neural networks are a machine learning algorithm modeled on our brains and designed to make decisions. Recommendation algorithms are popular and used by big businesses like YouTube and Netflix to predict what you like based on your previous choices.\(^\text{212}\)

Once you develop comfort with the syntax and structure of computer programming, algorithms will offer you a framework to solve complex problems and perform tasks efficiently.

**Object Oriented Programming**

Object-oriented programming (aka OOP) is a programming model that is based on the concept of classes and objects. A “class” in OOP represents the properties and behaviors of entities relevant to the program. Every time you create an instance of a class, you have an object with values attached to the predefined characteristics of the class. For example, if you are building a software to record student progress at a school, then Students and Teachers could be objects relevant to the program. Student objects would have properties such as names and addresses and behaviors such as course registration. Every time a new student or a teacher joins, you could create an instance of the class with the appropriate properties such as name and age.

OOP allows you to organize your code in a modular and reusable way by encapsulating data and behavior into objects. In OOP, objects have attributes (data) and methods (functions) to operate on that data. For example, a car object might have attributes such as color, make, and model, and methods such as accelerate, brake, and turn.

OOP provides several advantages over simpler programming paradigms such as procedural programming.

- **Simplicity:** OOP allows you to define variables that closely resemble the properties and behaviors of the real world.

- **Modularity:** OOP allows you to break your code into small, reusable modules (objects).

- **Encapsulation:** OOP allows you to hide the internal details of an object and expose only the necessary methods and attributes.

- **Inheritance:** OOP allows you to create new classes that are based on existing classes (parent

---


classes). The children inherit attributes and methods of the parent classes. For example, “Transfer Students” and “International Students” may inherit all the properties of regular “Students,” but may also have extra traits specific to Transfer and International students. This greatly simplifies application development.

**Objects**

As described above, an object is an instance of a class. An object consists of one or more properties that are key-value pairs defining the characteristics of the object, and methods, which are functions that define the behavior of the object.

Objects are a core feature of most programming languages used today, including JavaScript, Python, and Java. They allow us to encapsulate related data and functionality into a single unit, which can be more convenient and easier to work with than using separate variables and functions.

Here is an example of an object in JavaScript that represents a recipe:

```javascript
let recipe = {
    recipeName: “Lemon Orange Cake”,
    recipeSource: “Taste of Home”,
    recipeDetails: function () {
        return recipeName + “from” + recipeSource;
    }
};
```

In this example, we have defined an object named recipe that has three properties: recipeName, recipeSource, and recipeDetails. The recipeName and recipeSource have basic information about the recipe, while the recipeDetails property is a method that returns a string with the recipe name concatenated to the recipe source.

We can access the properties and methods of an object using dot notation or bracket notation. For example, to access the recipeName property of the recipe object, we can use the following code:

```javascript
console.log(recipe.recipeName); // Output: Lemon Orange Cake
```

To call the recipeDetails of the recipe object, we can use the following code:

```javascript
console.log(recipe.recipeDetails()); // Output: Lemon Orange Cake from Taste of Home
```

Objects are a powerful tool in programming and make it easy to organize and manipulate complex data structures. They are used extensively in all types of programming.

**Popular OOP Languages**

Here are a few examples of programming languages that support object-oriented programming.

- **Java** is the first object-oriented language that gained commercial popularity. Java continues
to be a popular OOP language that is used for building mobile apps, web apps, and enterprise systems. It is known for its platform independence, robustness, enterprise support, and extensive libraries. Online learning resources, inbuilt functions, and libraries make Java easy to learn and use.

- **Python** is a versatile general-purpose OOP language that is used for a wide range of applications, including data analysis, machine learning, and web development. It is known for its simplicity, readability, and versatility.

- **C++** is a powerful OOP language created by the notable computer scientist Bjorne Stroustrup. It is known for its efficiency and ability to interact with hardware and is used to build high-performance applications, including operating systems, game engines, and embedded systems. Almost all widely used applications on desktops such as web browsers, email clients, and office suites are built using C++. It requires greater developer knowledge about the internal workings of computers and, therefore, is considered to have a steeper learning curve than other languages.

- **C#** is a modern OOP language that supports emerging design practices. It is used to build mobile apps, web applications, cloud-based services, and games. It is known for its simplicity, type safety, and extensive libraries.

- **Ruby** is a flexible, dynamic OOP language that is used for building web applications, scripting, and automation. It is known for its readability, expressiveness, and the Ruby on Rails web framework. Rails introduced the concept of Model-View-Controller to develop web applications, which is now the basis for almost all web applications written in any language.

- **Swift** is a modern, fast OOP language that is used for building iOS, macOS, and watchOS applications. It is known for its safety, speed, and ease of use.

- **JavaScript** did not start out as an OOP language, but it has acquired a lot of OOP characteristics. It is a versatile language that can be used for both OOP and procedural programming styles. It supports OOP concepts such as encapsulation and abstraction and offers built-in objects and methods that can be used for OOP.

**Applications**

Computer applications created by developers can typically be used in three different versions; (1) downloaded to desktops, (2) downloaded to mobile devices, or (3) accessed through browsers. This has evolved over time. Personal computers (desktops and laptops) running Microsoft Windows and Apple MacOS were the first computing platforms used to run applications. These became common in offices in the 1980s and homes in the 1990s. In the 2000s, the Internet and World Wide Web allowed desktops to connect to the web and exchange information. People began using web-based applications to file taxes and pay their bills. Then in 2007, Apple introduced the iPhone and the IOS mobile platform and launched the era of mobile applications.

---


Each of these shifts in technology changed how we humans conduct our day-to-day lives and connect with others for fun and work. These shifts also changed how we access applications and the type and variety of available applications.

**Desktop Applications**

Desktop applications are computer programs that are installed and run on a personal computer or a local network. Word processors, spreadsheets, graphics editors, and audio and video editors are desktop applications. Typically, desktop applications are faster, more powerful, and offer more features than web applications. Desktop applications can also be used offline, without an Internet connection. Here are just a few examples of desktop applications:

- Word, Excel, PowerPoint, and Outlook from Microsoft are desktop applications widely used for business and personal productivity. Chrome, Safari, Edge, Firefox, and Opera are popular web browsers.
- Photoshop, Illustrator, InDesign, and other applications are part of the Adobe Creative Suite and are used for graphic designing and photo and video editing.
- QuickBooks accounting software is used by businesses to manage financial transactions and generate reports.
- AutoCAD is a computer-aided design (CAD) software used by architects, engineers, and designers to create 2D and 3D drawings and models.
- Visual Studio is an integrated development environment (IDE) used by developers to create software applications for Windows, Mac, and other platforms.

**Web Applications**

Web applications run on remote servers and are accessed through a web browser. Facebook, Amazon, Gmail, Instacart, and Netflix are all web applications we use in our daily lives. Typically, web applications are relatively simpler in design and functions compared to desktop applications, as they are unable to use all the capabilities of desktops such as notifications. Popular web applications include:

- Gmail is used for emailing. Google Drive and Google Docs are used for storing information, editing, and collaboration.
- Amazon, Instacart, and other online shopping platforms allow users to buy and sell products and services, browse reviews and ratings, and compare prices.
- Yelp, TripAdvisor, and other rating platforms let users review products and services before making a major purchase or committing to a hiring decision.
- Monday.com, Trello, and Asana are web-based tools that allow users to manage projects with the help of online boards, calendars, approval workflows, and reports.

Web applications are available 24*7 from any device that has a reliable Internet connection. Not surprisingly, they have become an essential part of our daily lives—we cannot imagine going back to the time when we didn’t have web applications.
Mobile Applications

Mobile applications run on mobile devices such as smartphones and tablets. In addition to the standard pre-installed apps that come with smartphones, you can buy and install mobile apps from mobile stores such as Microsoft Store, Google Play, and the App Store.

Mobile apps are currently designed for the two major mobile platforms in the market: Android and iOS. These apps enable much of social networking, gaming, entertainment, communication, and even productivity, particularly since remote work became popular after Covid.

- WhatsApp is a popular mobile app that transformed the world of voice and video calling by offering the services for free.
- Instagram, a photo and video sharing app, allows users to upload, edit, and share photos and videos.
- Spotify changed the world of music streaming by allowing users to listen to songs and to create playlists using their mobile phones.
- Google Maps not only provides directions but also real-time traffic updates and information on nearby businesses.

The ease of use of mobile apps has led to a rapid growth in the app development industry, with new apps being released every day. If programming is what you do for fun, you should consider building a mobile app and putting it for sale on online stores.

The Interplay Between Desktop, Web, and Mobile

As you spend more time in the technology world, you will notice an interplay between desktop, web, and mobile apps. In the morning, you may open a word document on your laptop and edit it using Word. As a copy of the document is saved to Microsoft’s servers, you may later log into a library computer to access the same document through the browser. On the train ride back home, you may open the same doc on your mobile app and forward it to a friend. Although you are interacting with the same document, you are using different interfaces based on your needs. Most popular applications such as Gmail, Google Maps, and Discord have desktop, mobile, and browser-based versions.

Providers that sell both computers and mobile phones are increasingly developing end-to-end ecosystems, so your phone calls, meetings, alerts, and chat messages interact with you seamlessly through your desktop and phone.

Computer Program Design

When you first decide to write a software program, you should spend a bit of time designing the program first. Above all, the program should be safe and should not expose internal company information or customers’ personal information to the outside world. The program should also be designed so it does not consume more database or server resources than necessary. The UI of the program must be attractive and easy to use if you do not want your program to fall in the huge garbage bin of discarded, unused programs.

Only after you are completely satisfied with the design, you should begin writing code in a language
best suited for the purpose. Most organizations have adopted an iterative programming model, so you will write small modules of the program, test it, and show it to potential users before going back to write more code.

The language you choose for your program will depend on the needs of the program, the resources available to you, and the platform where you want to deploy your program.

**Emerging Technologies**

There is an adage in the technology industry that technology helps us build better technology. If you observe how computer chips are made, you will notice that the most powerful chips we have were made possible because of technology. It would have been impossible to build today’s computer chips in the 1980s or 1990s. The pace of development of several other technologies has increased in recent decades. The power of the mobile phone, your ability to connect to a GPS satellite and ask for navigation, the cost/speed of sending money to a friend, or the cost of messaging your friends are all examples of this trend. Being aware of emerging technologies can help us anticipate how our future is likely to shape over the next few decades. We provide a brief overview of some emerging technologies that we believe will have the greatest impact in our lifetimes.

**Artificial Intelligence (AI) and Machine Learning (ML)**

Artificial Intelligence is technology that enables computers to perform tasks normally associated with human intelligence. Intelligent systems tend to mimic the human ability to interact with each other to solve problems using capabilities such as image recognition, speech recognition, and language comprehension. If you have interacted with Apple’s Siri or Amazon’s Alexa, you must have noticed that they can understand your verbal commands and respond with an answer in an audio form. Similarly, you can direct questions or commands at ChatGPT, which may seem like a back-and-forth conversation with another human. This ability of software programs and devices to simulate human intelligence is Artificial Intelligence (AI). Self-driving cars, Facebook programs that tag photos with names, and automated fraud detection by credit card companies are all applications of AI software.

Machine Learning is a subset of Artificial Intelligence that focuses on using large datasets to train models that can then predict outcome for new data sets. For example, you can use the data of all recently sold homes in a neighborhood to train a pricing model to figure out how homes are priced. Then, the model can be used to price new homes coming up on the market. Popular real estate pricing tools like Redfin and Zillow use Machine Learning to suggest possible rent and sale prices of apartments and homes. Machine Learning software can assist radiologists in reading X-rays, MRIs, CT scans, and Mammograms. ML is also widely used in voice recognition, email filtering, malware detection, fraud identification, and preference analysis.
Machine Learning and Artificial Intelligence are highly likely to change our lives. While there is great fear that these technologies may make some jobs redundant, they will also create new job categories that we don’t even know about. For example, Figure 229 shows that a large percent of jobs done today are new, and this trend is likely to continue.

However, we must exercise caution when adopting AI and ML models because the long-term impact on humans is difficult to predict and the regulatory framework around these technologies, to guide software makers is still in its infancy.

---

Blockchain and Cryptocurrencies

Bitcoin and other cryptocurrencies are based on an underlying technology called Blockchain. It is a decentralized digital ledger that records transactions in a secure and transparent manner. A collection of computers, owned by individuals all over the world, can run a blockchain software that enables them to interact with one another, review transactions, and add to the common ledger. Because no single computer controls the ledger, and all computers must agree before a ledger can be updated, the transactions are considered reliable and secure.

While cryptocurrencies are the most prominent use of blockchains, the underlying technology can be used in other areas, too. Supply chain management, voting record management, and digital identity management are a few such fields.

Ethereum, a variation of the Bitcoin blockchain, enables you to not only track data in a distributed ledger but also execute code. This lets developers build distributed apps on the Ethereum blockchain where “smart contracts” can be written into the code. The contracts execute automatically when conditions are met. For example, if you implement a supply chain system on the Ethereum blockchain, your supply chain app could be written with smart contracts to automate payment to the vendor when the shipment has reached its destination. When the ledger is updated with the “shipment received” event, the money is automatically moved from the buyer’s wallet to the seller’s wallet, without any human intervention. This greatly reduces the transaction cost, improves transparency, and avoids risk.

Augmented Reality and Virtual Reality

Augmented Reality (AR) combines digital information and real-world information in one place. If you are a surgeon with AR glasses, you would have vital information about the patient as you look at the patient on your operating table. A warehouse worker can wear glasses that show him directions to the precise location of an item in the warehouse. If you are a student reading a textbook, you may want additional information about the topic visible to you as you read. If you are looking at a toy in Walmart, AR can display special offers and user reviews about that product.

AR uses cameras, computers, displays, computer vision, and context-aware software to project information that appears to overlay the real world. Microsoft’s HoloLens is an example of an AR device.216

Virtual Reality (VR) focuses on simulating new reality. The real world is blotted out, and users are immersed in a computationally generated world.

---

In the military, VR is used to train soldiers by simulating flight and battlefield scenarios. In medicine, students use VR to study the human body and surgical procedures. Vision Pro VR headset from Apple provides an immersive gaming and movie watching experience.

VR generally uses a headset as it needs to immerse users in a simulated world. You don’t need headsets to experience AR. Cars with AR can display directions and local traffic information overlaid on the screen display or windshield. This means that as you navigate your car, you have information in front of your eyes to help you make judicious decisions without taking your eyes off the road.

Internet of Things

Devices connected to the Internet form the Internet of Things (IOT). Your old doorbell did a good enough job of notifying you when someone rang the bell. However, if you attach a camera and a microprocessor to it and connect it to the Internet, it can notify you on your mobile device when someone is at your doorstep. It is not only a doorbell but also a security and communication device that lets you interact with the person at the door (even if you are not at home) to receive a package or let a family member in. Ring and Nest have become popular by selling smart doorbells. If you own a fleet of trucks, you can connect smart sensors to your trucks to notify you ahead of time about any required maintenance. Your smart fridge may determine that you are short on milk, eggs, and vegetables and place an order on your behalf for online delivery.

The Internet of Things and devices connected to the Internet can lead to savings of time and energy. The world as we know it will change as a result because many of the manual chores that we have been doing for ages will be done automatically by Internet connected devices.

Autonomous Vehicles (Self-Driving Cars, EVTOL, Drone Delivery)

If you live in San Francisco, Phoenix, LA, Beijing, or Las Vegas, you can hail a self-driving car. A recent McKinsey article estimates that a sizeable chunk of new cars will have autonomous driving features by 2035. The report predicts that self-driving cars will lead to fewer accidents, increase in productivity, and improve mobility for senior drivers.

The flying car has been promised by many science fiction writers, but it is not here yet. However, don’t be surprised if, within a decade, you look up at the sky and see streams of flying vehicles autonomously

---

delivering people and goods from point A to B. Companies like Joby aviation,\textsuperscript{218} Lilium,\textsuperscript{219} and Archer Aviation\textsuperscript{220} have prototype aircrafts that are already going through FAA certification to deliver that dream.

Developments in battery technologies, computer vision, electric motors, and robotics have made it possible to design drones that are whisper quiet, safe, and cheap. It is expected that by 2030, for about what you pay for an uber ride, you can catch an EVTOL (Electric Vertical Take Off and Landing) drone from New York JFK airport to downtown Manhattan or from Chicago O’Hare to downtown Chicago.

**Advanced Robotics**

Imagine a world where a robot follows you around carrying your coffee and your laptop. A robotic dog walks around your property guarding your perimeter, looking for trespassers, broken fences, navigating difficult terrain, and using its camera like eyes. It can identify security risks and let you know if anything is amiss. Imagine a robotic exo-skeleton that lets a soldier carry 100s of pounds with ease over difficult terrain. A robotic surgeon that can do surgery with precision. A drone that can act as a refueling gas station in the sky for military aircraft. Robots that can run into a fiery building and rescue people.

All these science-fiction sounding robots are available today. And they are continually getting better as their underlying computers, batteries, motors, and machine learning improve.

**Biotechnology**

Emerging biotechnology trends have the potential to impact our health and happiness in a big way. Here are a few:

- Gene editing techniques such as CRISPR have revolutionized the field of genetics and have the potential to cure genetic diseases. In the future, gene editing could become more precise, efficient, and accessible, opening new possibilities for genetic therapies.

- Personalized medicine is becoming a reality with advances in genomics and data analytics. Healthcare providers can use genetic data to develop targeted therapies for individual patients, reducing the need for trial and error in treatment and improving patient outcomes.

- Synthetic biology involves the design and engineering of biological systems for specific purposes. This field has the potential to create new biologically inspired materials, improve food production, and develop novel treatments for diseases.

- Cell therapies involve using living cells to treat diseases. Stem cell therapies are already being used to treat some diseases, but researchers are exploring the potential of other cell types, such as immune cells, to treat cancer and other conditions.

\textsuperscript{220} Archer homepage, \url{https://www.archer.com/} (accessed June 2023).
Microbiome research involves the human microbiome, which consists of trillions of microorganisms living in and on the body, and is increasingly recognized as playing a crucial role in human health. Researchers are studying the microbiome to better understand its impact on various diseases and to develop new therapies.

Overall, biotechnology is a rapidly evolving field, and new breakthroughs are likely to continue to emerge in the coming years. These advances have the potential to revolutionize not just medicine but also agriculture and other industries, improving our overall quality of life.

**Learning to Program: Getting Started**

Making programming easy for newcomers is a priority for the industry. We provide some recommendations here to get you started. As you develop comfort, continue to search online for additional or superior resources to grow your skills.

Most experts agree that Python is the best language to start learning programming. For this reason, most college courses, even at the most selective colleges, use Python for the introductory programming class.

The simplest way to get started is to use an online environment. Jdoodle lets you get started without creating an account. By clicking a button, you can run the basic addition program. You can access it at: [https://www.jdoodle.com/python3-programming-online/](https://www.jdoodle.com/python3-programming-online/).

If you are willing to create an online account, you can also use replit. REPL stands for Read-Eval-Print-Loop, which is any interactive interface that can accept a user program as input, execute (evaluate) it, print the result for the user to check, and be ready for the next program (loop). Replit is a popular modern commercial implementation. You can access the Python interpreter at Replit—[https://replit.com/languages/python3](https://replit.com/languages/python3).

Once you settle on an environment, you need a lesson plan. The official Python tutorial is at: [https://docs.python.org/3/tutorial/](https://docs.python.org/3/tutorial/).

The official Python tutorial is text-heavy. For a more visual experience, YouTube has a few extremely popular introductory Python tutorials. “Learn Python—Full Course for Beginners” by Mike Dane is one of the most popular. You can get started by practicing these exercises on either JDoodle or Replit. Once you get comfortable and if you have access to a computer on which you can install software, you can install Python on your computer and write your programs locally.

In 2023, we also have an excellent resource to learn programming—ChatGPT. You can ask ChatGPT to write programs to meet your specifications as shown in Figure 230 (top). After you complete a few initial lessons in the tutorial of your choice, we encourage you to explore other similar examples from ChatGPT. To use ChatGPT, go to the website [https://chat.openai.com](https://chat.openai.com) and sign up for an account. After that, the site is as intuitive to use as a search engine.

---

221 “Python—Full Course for Beginners,” on YouTube at [https://www.youtube.com/watch?v=rfscVS0vtbw](https://www.youtube.com/watch?v=rfscVS0vtbw) (accessed June 2023).
Certainly! Here’s a simple Python program that performs basic arithmetic operations with numbers:

```python
# Perform arithmetic operations
result1 = 10 + 5
result2 = 10 - 5
result3 = 10 * 5
result4 = 10 / 5

# Display results
print("Result 1:", result1)
print("Result 2:", result2)
print("Result 3:", result3)
print("Result 4:", result4)
```

In this program, we perform addition, subtraction, multiplication, and division operations using numbers directly in the arithmetic expressions.

The results of these operations are stored in separate variables: `result1`, `result2`, `result3`, and `result4`.

Finally, we use the `print()` function to display the values of the results.

When you run this program, it will output the results of the arithmetic operations.

What is the error in this Python program?
```python
result4 = 10 /+ 5
print("Result 1:", result1)
```

The error in the given Python program is a syntax error in the line `result4 = 10 /+ 5`.

The `+/` operator combination is not a valid operator in Python. To perform addition and division, you need to separate the operators and use them individually.

To fix the error and perform division, you should use the `/` operator alone, without the `+` symbol. Here’s the corrected version of the program:

```python
# Perform arithmetic operations
result4 = 10 / 5

# Display result
print("Result 4:", result4)
```

In this corrected version, the program correctly calculates the division of 10 by 5 using the `/` operator and assigns the result to `result4`. Finally, it prints the value of `result4`.

FIGURE 230 — ChatGPT can be used to learn and write code (top). Then Chat GPT can be asked to evaluate and help debug code (bottom).

You can also use ChatGPT to identify errors in your programs as shown in Figure 230 (bottom). ChatGPT will not just fix errors but also explain them. ChatGPT can, therefore, serve as your personal tutor and debugger as you write programs.
Once you get comfortable with basic programs and are curious to see how these programs work, Python Tutor is an excellent resource to show you exactly what happens inside the computer as it processes each line of the program. Many of the top computer science programs use Python Tutor in introductory programming classes. You can access Python Tutor at https://pythontutor.com/.

Writing programs is thrilling, and the 2020s are a great time to learn computer programming. Between the online REPL environments, structured tutorials, and ChatGPT, all of which are available for free, you have everything to develop proficiency in computer programming at your own pace.
Algorithm: A set of instructions for solving a problem

Augmented Reality (AR): Combines digital information and real-world information in one place

Compiler: A specialized software program that can convert software code written from specific languages into binary code, which can then be executed directly by the computers

Domain-Specific Languages: Computer languages optimized for specific application domains; examples include SQL, R, and MATLAB

Internet of Things (IOT): Devices that serve a specific purpose and are connected to the Internet

Low-Level Languages: Programming languages that are close to the processor’s native instruction set; these languages are sometimes called assembly language

Machine Learning: A subset of Artificial Intelligence that focuses on using large datasets to train models that can then predict outcome for new data sets

Markup Languages: Computer languages used to specify how information should be displayed or interpreted; examples include HTML, Markdown, and XML

Object-Oriented Programming Languages: Computer languages that allow developers to create their own data types by organizing data and related functions into objects; examples include Java, C#, and C++

Procedural Programming Languages: Computer languages that use precise steps to compose programs; examples include C, Fortran, and Pascal

Scripting Languages: Computer languages used to automate tasks using the capabilities of existing applications; examples include AppleScript, JavaScript, and Python

Variable: A named storage location in a computer’s memory that holds a value
Chapter Case

Vivian’s Raspberry Pi

Vivian had a hard time with volleyball practice today. She couldn’t get her mind off of her science fair project idea. She wasn’t sure exactly what she wanted to do. All she knew is that she wanted to use a Raspberry Pi. Suddenly…BOOM! Here comes the volleyball straight at her head! And then…she got an idea!

A Raspberry Pi is a small low-cost computer that enables people of all ages to explore technology and learn how to program. There are lots of projects students can explore such as building a robot or creating wearable technology. Here is a list of projects: https://projects.raspberrypi.org/en. The projects are designed for people all over the world who speak different languages. For instance, here is the link for projects with the instructions in Spanish: https://projects.raspberrypi.org/es-ES. The whole world can have fun with Raspberry Pi!

Question 1: Look through the projects on the Raspberry Pi website. Find a project that interests you. Describe at least 2 things about the project that interests you to include: What will you make and what will you learn?

Question 2: Each Raspberry Pi project includes a detailed set of Instructions. Look through all the steps included in the instructions of your project. What do you think would be the most difficult task for you to complete and why do you think it would be so difficult? Now, describe what you think you would need to learn in order to complete this most difficult task.