INTERACTION DESIGN

# PROGRAMMING 101

Creative Coding HS21

# PROGRAM

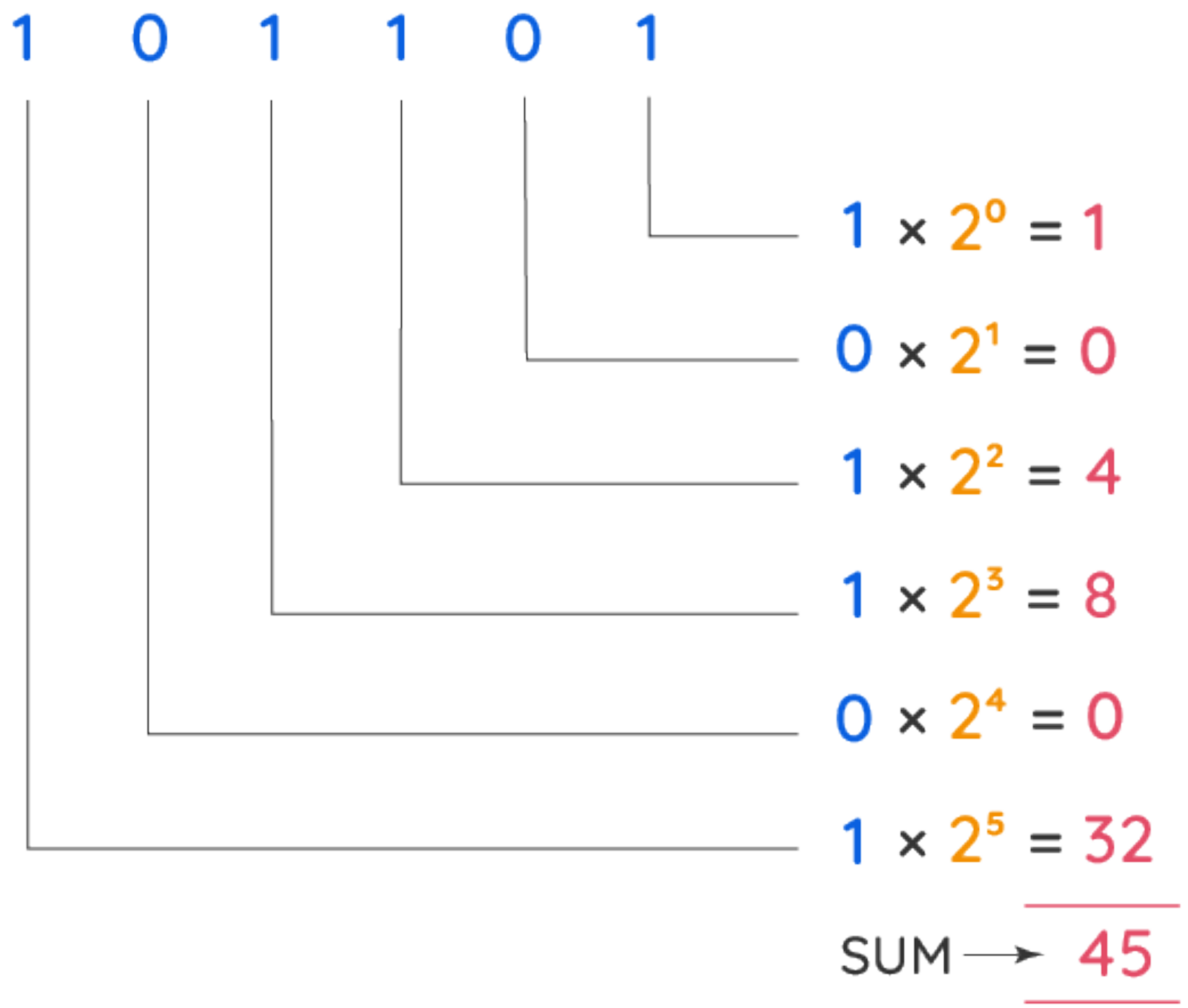
A set of instructions is called a **program,** whose task is to tell the computer what to do or tell other people what we want the computer to do.

# COMPUTER LANGUAGE

Data and instructions (numbers, characters, strings, etc.) are encoded as binary numbers - a series of bits (**1**s or **0**s). Encoding and decoding of data into binary is performed automatically by the system based on the encoding scheme. **The digits 1 and 0 used in binary reflect the on and off states of a transistors** that a computer processor (CPU) is made of.

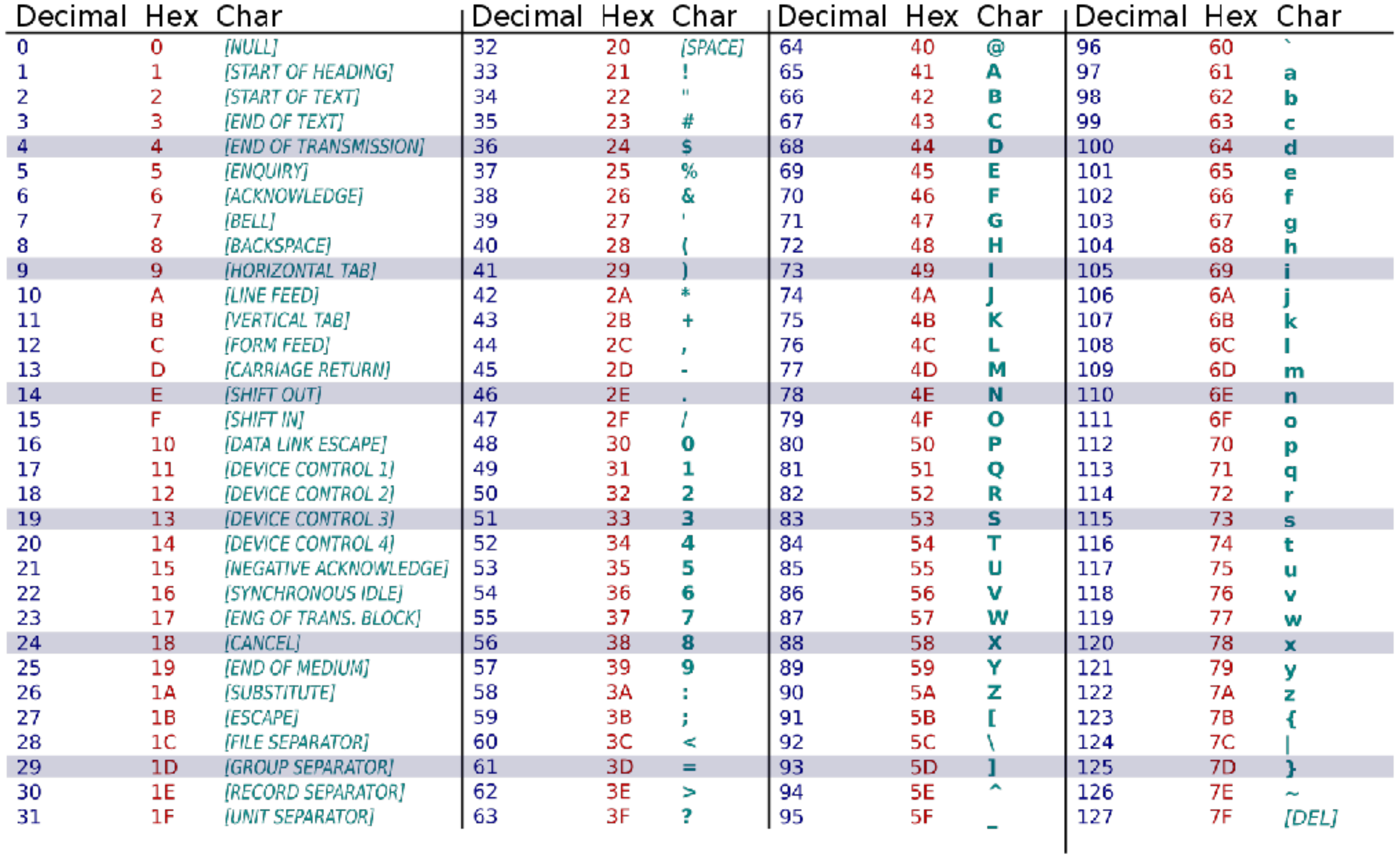
# COMPUTER LANGUAGE

Numeric Data is encoded as binary numbers.



# COMPUTER LANGUAGE

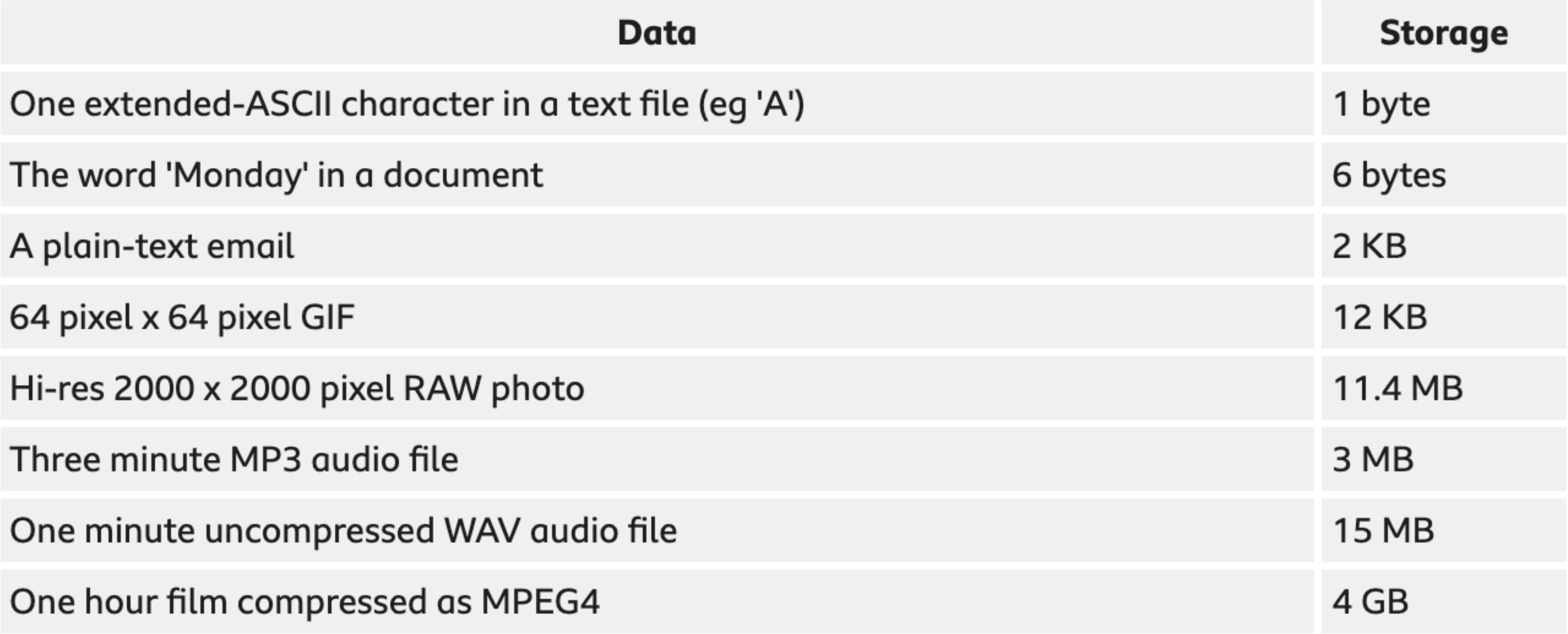
Non-Numeric Data is encoded as binary numbers using representative code, either **ASCII** – 1 byte per character or Unicode – 2 bytes per character.



| Decimal | Hex | Char | Decimal | Hex | Char | Decimal | Hex | Char | Decimal | Hex | Char |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | [NULL] | 32 | 20 | [SPACE] | 64 | 40 | @ | 96 | 60 | ` |
| 1 | 1 | [START OF HEADING] | 33 | 21 | ! | 65 | 41 | A | 97 | 61 | a |
| 2 | 2 | [START OF TEXT] | 34 | 22 | " | 66 | 42 | B | 98 | 62 | b |
| 3 | 3 | [END OF TEXT] | 35 | 23 | # | 67 | 43 | C | 99 | 63 | c |
| 4 | 4 | [END OF TRANSMISSION] | 36 | 24 | $ | 68 | 44 | D | 100 | 64 | d |
| 5 | 5 | [ENQUIRY] | 37 | 25 | % | 69 | 45 | E | 101 | 65 | e |
| 6 | 6 | [ACKNOWLEDGE] | 38 | 26 | & | 70 | 46 | F | 102 | 66 | f |
| 7 | 7 | [BELL] | 39 | 27 | ' | 71 | 47 | G | 103 | 67 | g |
| 8 | 8 | [BACKSPACE] | 40 | 28 | ( | 72 | 48 | H | 104 | 68 | h |
| 9 | 9 | [HORIZONTAL TAB] | 41 | 29 | ) | 73 | 49 | I | 105 | 69 | i |
| 10 | A | [LINE FEED] | 42 | 2A | * | 74 | 4A | J | 106 | 6A | j |
| 11 | B | [VERTICAL TAB] | 43 | 2B | + | 75 | 4B | K | 107 | 6B | k |
| 12 | C | [FORM FEED] | 44 | 2C | , | 76 | 4C | L | 108 | 6C | l |
| 13 | D | [CARRIAGE RETURN] | 45 | 2D | - | 77 | 4D | M | 109 | 6D | m |
| 14 | E | [SHIFT OUT] | 46 | 2E | . | 78 | 4E | N | 110 | 6E | n |
| 15 | F | [SHIFT IN] | 47 | 2F | / | 79 | 4F | O | 111 | 6F | o |
| 16 | 10 | [DATA LINK ESCAPE] | 48 | 30 | 0 | 80 | 50 | P | 112 | 70 | p |
| 17 | 11 | [DEVICE CONTROL 1] | 49 | 31 | 1 | 81 | 51 | Q | 113 | 71 | q |
| 18 | 12 | [DEVICE CONTROL 2] | 50 | 32 | 2 | 82 | 52 | R | 114 | 72 | r |
| 19 | 13 | [DEVICE CONTROL 3] | 51 | 33 | 3 | 83 | 53 | S | 115 | 73 | s |
| 20 | 14 | [DEVICE CONTROL 4] | 52 | 34 | 4 | 84 | 54 | T | 116 | 74 | t |
| 21 | 15 | [NEGATIVE ACKNOWLEDGE] | 53 | 35 | 5 | 85 | 55 | U | 117 | 75 | u |
| 22 | 16 | [SYNCHRONOUS IDLE] | 54 | 36 | 6 | 86 | 56 | V | 118 | 76 | v |
| 23 | 17 | [ENG OF TRANS. BLOCK] | 55 | 37 | 7 | 87 | 57 | W | 119 | 77 | w |
| 24 | 18 | [CANCEL] | 56 | 38 | 8 | 88 | 58 | X | 120 | 78 | x |
| 25 | 19 | [END OF MEDIUM] | 57 | 39 | 9 | 89 | 59 | Y | 121 | 79 | y |
| 26 | 1A | [SUBSTITUTE] | 58 | 3A | : | 90 | 5A | Z | 122 | 7A | z |
| 27 | 1B | [ESCAPE] | 59 | 3B | ; | 91 | 5B | [ | 123 | 7B | { |
| 28 | 1C | [FILE SEPARATOR] | 60 | 3C | < | 92 | 5C | \ | 124 | 7C | \| |
| 29 | 1D | [GROUP SEPARATOR] | 61 | 3D | = | 93 | 5D | ] | 125 | 7D | } |
| 30 | 1E | [RECORD SEPARATOR] | 62 | 3E | > | 94 | 5E | ^ | 126 | 7E | ~ |
| 31 | 1F | [UNIT SEPARATOR] | 63 | 3F | ? | 95 | 5F | _ | 127 | 7F | [DEL] |

# COMPUTER LANGUAGE

A group of 8 bits is called a byte. Most computers can process millions of bits every second.



| Data | Storage |
|---|---|
| One extended-ASCII character in a text file (eg 'A') | 1 byte |
| The word 'Monday' in a document | 6 bytes |
| A plain-text email | 2 KB |
| 64 pixel x 64 pixel GIF | 12 KB |
| Hi-res 2000 x 2000 pixel RAW photo | 11.4 MB |
| Three minute MP3 audio file | 3 MB |
| One minute uncompressed WAV audio file | 15 MB |
| One hour film compressed as MPEG4 | 4 GB |

# COMPUTER LANGUAGE

Computers can not use human languages, and programming in the binary language of computers is a very difficult, tedious process. Therefore, most programs are written using a programming language and are converted to the binary language used by the computer.

# COMPUTER LANGUAGE

There are three major categories of programming languages:

1. Machine Language
2. Assembly Language
3. High level Language

# ASSEMBLY LANGUAGE

English-like abbreviations are used for operations

```
        global    _start

          section   .text
_start:   mov       rax, 1                  ; system call for write
          mov       rdi, 1                  ; file handle 1 is stdout
          mov       rsi, message            ; address of string to output
          mov       rdx, 13                 ; number of bytes
```

Was developed to make programming easier.

Assembler is used to convert assembly language programs into machine code

# High Level Languages

English-like and easy to learn and program

Common mathematical notation

e.x Java, C, C++, Visual Basic

# High Level Languages

A program written in a high-level language is called a *source program (or source code)*. Since a computer cannot understand a source program. Program called a *compiler* is used to translate the source program into a machine language program called an *object program*. The object program is often then linked with other supporting library code before the object can be executed on the machine.

# High Level Languages

The majority of the most popular programming languages are what’s known as object-oriented. Essentially, this is a model of programming that’s organised around data (also called objects). **OOP** languages are easy to manipulate, reuse, and scale, making them ideal for structuring software.

# Object Oriented Programming

## Class

Specifies the definition of a particular kind of object

**Characteristics:**
Properties (or Attributes)

**Behaviors:**
Methods used as a template to create objects of that type

## Object/Instance

An object created using the Class Definition

All specific instances of the same class share the same definition

# Object Oriented Programming

```
Class Cats {
  constructor() {
    name
    breed
    age
    weight
  }
    meow()
    sleep()
    eat()
    hunt()
  }
```

```
myCat = new Cat() {

  name = Lucy
  breed = Persian
  age = 3 years
  weight = 5.2kg

  meow()
  sleep()
  eat()
}
```

# Object Oriented Programming

```
Class Cats {
  constructor() {
    name
    breed
    age
    weight
  }
    meow()
    sleep()
    eat()
    hunt()
  }
```

```
myCat = new Cat() {

  name = Lucy
  breed = Persian
  age = 3 years
  weight = 5.2kg

  meow()
  sleep()
  eat()
}

  myCat.meow();
  myCat.sleep();
```

# C#

**C#** is a programming language designed by Microsoft. It was designed to improve upon existing concepts in C (procedural computer programming language).

- Microsoft .NET framework
- Web apps & mobile apps
- Games (Unity)

# JAVA

**Java** was developed in 1995 and gained popularity very fast because once you write a piece of code in Java, it can run on just about any device with the Java platform.

- Business software
- Web applications
- Mobile apps (Android)

# PYTHON

**Python** is a high-level programming language launched in 1992. It’s built in such a way that it’s relatively intuitive to write and understand, making it ideal for those who want rapid development.

- Web applications
- Artificial intelligence and machine learning
- Visual programming (plugins)

# RUBY

**Ruby** was conceived in 1993 as a general-purpose OOP. It became popular once Ruby on Rails was released (a web application framework written in Ruby)

- Web-applications
- Data analysis
- Prototyping

# JAVASCRIPT

**JavaScript** was developed in 1995 as a language that runs inside a client browser and processes commands on a computer rather than a server. It is commonly placed into an HTML or ASP file.

- Web and mobile applications
- Interactive applications
- Visual Coding

# WHY JAVASCRIPT?

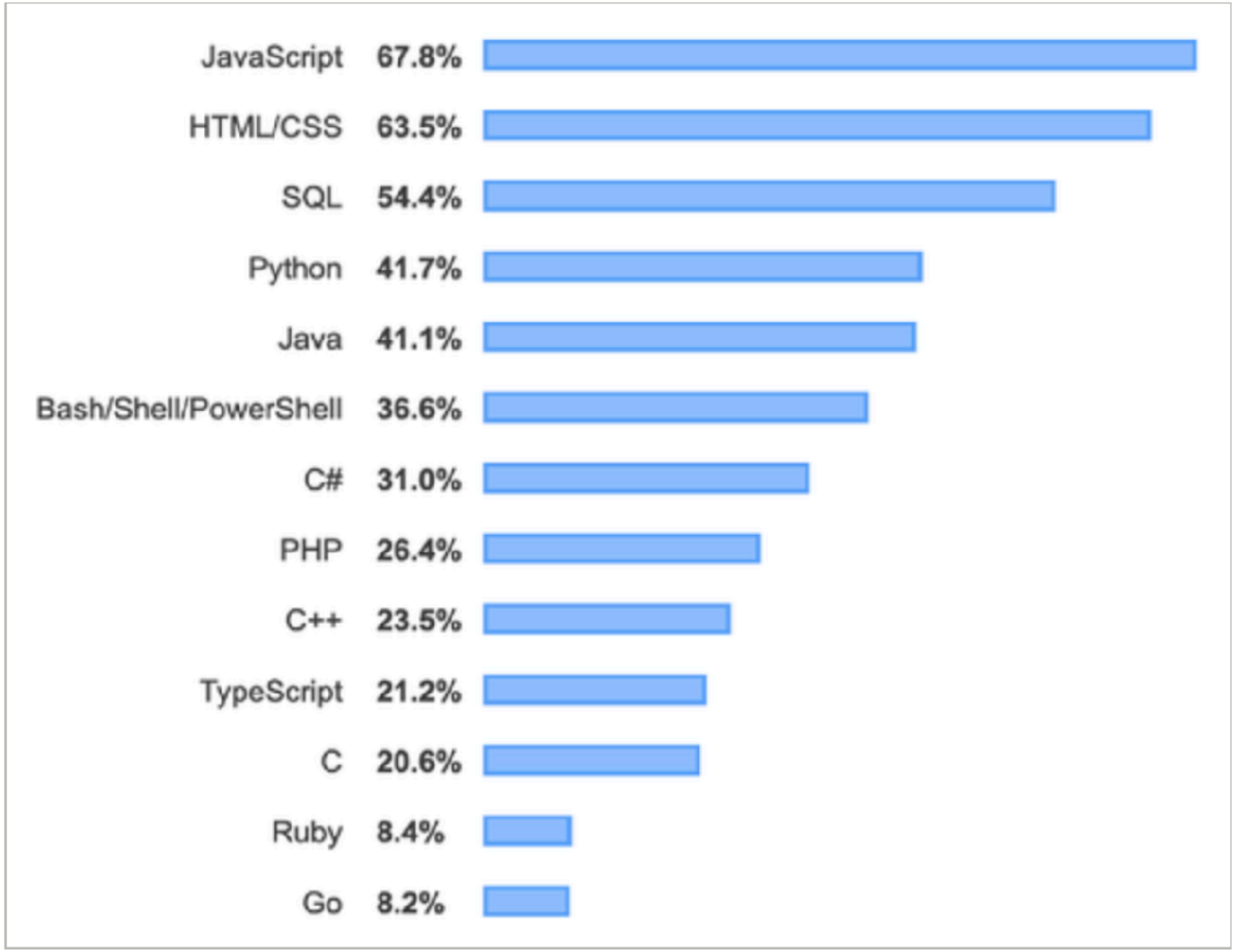


**FIGURE 1** Open in figure viewer | PowerPoint

Most popular programming languages on Stack Overflow, 2019

# What’s the difference between **JavaScript** and **P5.js** ?

https://github.com/sorrycc/awesome-javascript

# IDE

Web editor
Openprocessing
Processing IDE
Code Editor (VSCode, Atom, Brackets etc.)

# SYNTAX

JavaScript is case-sensitive. You can start a variable with be a letter (a-z, or A-Z), an underscore(_), or a dollar sign ($). It is a good practice to use camel case for the identifiers, meaning that the first letter is lowercase, and each additional word starts with a capital letter.

```
var clickCounter
var redirectPage123
var amountInArray$
```

# SYNTAX

JavaScript provides different datatype to hold different values on variable. JavaScript is a dynamic programming language, it means do not need to specify the type of variable.

```
// It store string data type
var txt = “Creative Coding";

// It store integer data type
var a = 5;
var b = 5;

// It store Boolean data type
(a == b )

// It store array data type
var places= ["GFG", "Computer", "Hello"];

// It store object data
var Student = {firstName:"John", lastName:"Doe", age:19, skill:"prototyping"}
```

# SYNTAX

JavaScript supports both single-line and block comments.

```
// this is a single-line comment

/*
* This is a block comment that can
* span multiple lines
*/
```

# SYNTAX

JavaScript does not require to end a statement with a semicolon (;), but it is recommended to always use the semicolon to end a statement.

```
var clickCounter;
var redirectPage123;
var amountInArray$;
```

# SYNTAX

You can use arithmetic operation with any type of data.

```
let number1 = 2;
let number2 = 33;
let value = 5;
```

**Addition:** number1 + number2 //35

**Subtraction:** number2 - number2 // 31

**Multiplication:** number1* number2 // 66

**Division:** number2 / number1 // 16.5

**Remainder:** number1 % number2 // 2

**Increment:** ++value, value++ // 6

**Decrement:** --value, value— // 4

**Negate:** -value // -5

**Convert to number:** +value. //5

# SYNTAX

Function is block of code designed to perform a particular task and it’s a core of JavaScript language. You call a function

```
function makeCoffe() {
  var nextCoffee = new Coffee(); // Function code
  nextCoffee.brew(frenchPress, arabica);
}

function brew(brewType, beanType) { //function with two arguments
  coffee = brewType + beanType
  return coffee;  ends function execution and specifies a value to be returned to the
  function caller
}
```

# SYNTAX

The **console.log()** method outputs a message to the web console. The message may be a single string (with optional substitution values), or it may be any one or more JavaScript objects.

```
console.log(‘Hello, world!’);

// you can use console.error to printing error information to standard error
console.error('Something went wrong!');
```

# Programming

The core part of learning to program is learning how to think about arranging the sequence of instructions to solve the problem or carry out the task.

# Polya’s 4 Steps of Problem Solving

**Understand the Problem**

Do I completely understand what is being asked ?

What are the requirements, what is expected ?

**Devise a Good Plan to Solve the Problem**

Develop your Algorithm – AT LEAST ONE

**Implement the Plan**

Follow through with your algorithm

**Evaluate the Implementation**

Did it work?

Did it solve the problem correctly and completely?

Is there another way to solve the problem?

George Polya – 1950’s