

T1A1 WORKBOOK (Timothy Long CAB022105)

Q1 Five Key events in the development of the internet

There are plenty of major events in the history of the internet, which guided the development of what we have today. This is an important point, as the development is not exactly linear, with a start to finish trajectory; it is widely expansive and developments can push the internet into directions that are exciting and often unpredictable. This discussion focuses on are some of the main events from 1980 that had enduring affects on the system we have now.

1) 1989/1990 Tim Berners-Lee creates World Wide Web

It would be somewhat remiss not to include the creation of the World Wide Web in this answer, given its primary significance in the development of the internet. Credit for this invention is given to Sir Tim Berners-Lee, a scientist working at CERN, who invented the the World Wide Web to share information between universities and other global institutes (1). His first proposal was written in 1989 and by the end of 1990 the first server and browser was running at CERN (2). This was a massive step in utilising the internet to share information between colleagues, and with assistance in its development, including from Nicola Pellow during her student placement, the software became more widely available to those using CERN computers.

By this stage, with the ball rolling, and additional contributions that Berners-Lee himself pled for (1), the software was able to scale in development and access, promoting use by the public in 1993 (1).

This development is not only instrumental in the creation of resources to share information via the web, but illustrative of the philosophy of the internet; to promote wide-ranging and free access to information. Today, Sir Tim Berners-Lee remains Director of W3C (World Wide Web Consortium) which develops standards and guidelines to lead the web to its full potential and remain accessible across multiple technologies (2).

1. CERN, *Where the Web was born*, viewed 18/09/2021, <https://home.cern/science/computing/birth-web/short-history-web>
2. W3, Tim Berners-Lee, *Biography*, viewed 18/09/2021, <https://www.w3.org/People/Berners-Lee/>

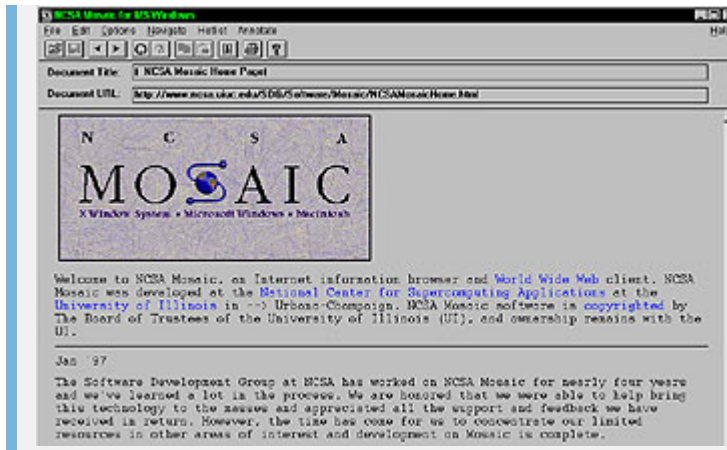
2) 1993 Creation of Mosaic (first web browser combining graphics)

Shortly after the World Wide Web was created, a web browser was launched that is credited to popularising access to the web; Mosaic (1). It was not the first browser, but it had an accessible and reliable interface, with simple installation, and user friendliness suited to the general public (1). Given that existing browsers tended to be difficult to use, and ran on expensive Unix machines, the inventors Marc Andreessen and Eric Bina worked to create a change (1).

Indeed, Moscaic is considered to be the first easy-to-use browser (2), and comparatively their browser was graphically superior, supported additional formatting options for HTML (such as center) and included the image tag (1). Instantly, this allowed integration of images within the webpage, whereas previously images

were only accessible as separate files (1). Additional features included the ability to easily scroll through text, and supported clickable hyperlinks, as opposed to the old system of users typing hyperlink reference numbers for re-direction (1).

These features improved user experience significantly, which can be opined to have contributed to the successful uptake of internet browsing by the public. Indeed, Mosaic is considered as the browser that spearheaded the way into the 90's internet boom (3); a reminder that the success of technology can be highly dependant not just on its potential, but also on its accessibility.



Source (4)

With free downloads, the browser experienced high levels of growth, and support for PC and Mac versions was developed; advancing out of the scope of Unix only machines (1). By mid-1994 it was noted that downloads were around 50,000 per month, which is very impressive at that time of internet development (2). As the number of web users increased, the popularity of information sharing across the internet was able to gain significant momentum. Within 18 months post-inception, Mosaic had over a million users and facilitated the exponential growth of web servers and surfers (4).

Following this success, the creators of Mosaic went on to later pursue a commercial project; the Netscape Navigator browser (2).

1. 2021, History Computer, *Mosaic Browser*, viewed 18/09/2021, <https://history-computer.com/software/mosaic-browser-history-of-the-ncsa-mosaic-internet-web-browser/>
2. Vaughan-Nichols, S., 2018, ZDNet, *Mosaic turns 25: The beginning of the modern web*, viewed 18/09/2021, <https://www.zdnet.com/article/mosaics-birthday-25-years-of-the-modern-web/>
3. Sack, H., 2018, SciHi Blog, *How Mosaic has Changed the World*, viewed 18/09/2021, <http://scihi.org/ncsa-mosaic-web/>
4. Hart, D., 2004, National Science Foundation, *Mosaic Launches an Internet Revolution*, viewed 18/09/2021, https://www.nsf.gov/discoveries/disc_summ.jsp?cntn_id=100274

3) 1998... Google Inc... or something else ?

So, yes, admittedly the official birth of Google Inc. in this year (1) is an absolutely substantial part of the development of the internet. However, this answer will actually focus on another significant event in this year; the Antitrust Lawsuit against Microsoft by the Department of Justice, and the Attorney Generals of twenty states (2).

At the time, Microsoft was the world's most dominant software company, and was protecting a monopoly on not only its operating system but also its browser, Internet Explorer (2). The case questioned the legality of its anti-competitive techniques, such as against competitor Netscape Navigator, and also revealed that Microsoft considered the internet to be a threat to its market position, and sought to control it (2).

After losing the case, Microsoft were then somewhat incentivised to allow an opening in the market for the growth of other companies (2), even after overturning the decision following an appeal (3). It allowed a foot in the door for other companies, even those that would go on to create alternative browsers, and also created an agreement for Microsoft to share computing interfaces with competitors (3).

While overturned, the success of the case is measured in that it created an opportunity for greater innovation in the tech industry without being competitively crushed, and it is opined that this ironically benefitted such groups as Google, Amazon and Facebook, which have themselves become titans of the industry (2). Indeed, Microsoft's browsers Edge and Internet Explorer have measured at an approximate combined market share of less than 10% today while Google Chrome has been measured at an approximate market share of 65% (4).

Today, the case may be reflected upon when considering if the same principle of tech monopolies shouldn't also apply to some of the current tech giants. For instance, one might consider the instance that when questioned by American Senator Lindsey Graham, the founder of Facebook, Mark Zuckerberg, struggled to adequately or specifically answer the question "Who's your biggest competitor?" (5)

1. *From the garage to the Googleplex*, viewed 18/09/2021, <https://about.google/our-story/>
2. Blumenthal, R., Wu, T., 2018, The New York Times, *What the Microsoft Antitrust Case Taught Us*, viewed 18/09/2021, <https://www.nytimes.com/2018/05/18/opinion/microsoft-antitrust-case.html>
3. Beattie, A., 2021, Investopedia, *Why Did Microsoft Face Antitrust Charges in 1998?*, viewed 18/09/2021, <https://www.investopedia.com/ask/answers/08/microsoft-antitrust.asp>
4. Stat Counter Global Stats, *Browser Market Share Worldwide*, viewed 18/09/2021, <https://gs.statcounter.com/browser-market-share>
5. Jeong, S., 2018, The Verge, *Zuckerberg struggles to name a single Facebook competitor*, viewed 18/09/2021, <https://www.theverge.com/2018/4/10/17220934/facebook-monopoly-competitor-mark-zuckerberg-senate-hearing-lindsey-graham>

4) 1999 Napster

At around this time, an incredibly important event in both music innovation and the internet arose, which challenged not only the traditional music industry, but gave users a platform to directly share files, and forecast the future 'cloud' internet (1). This was Napster. It was a file sharing platform created by two teenagers, Sean Parker and Shawn Fanning (2) which in itself spoke of the nature of the internet, with its ability to give growth to strong ideas, regardless of the age of the creators.

Napster created the ability for music to be downloaded, for free, in a more reliable, consistent way with the concept of users sharing their MP3 music files (2) across the internet. By the following year, there were 20 million in the Napster community, and it gained the unfavourable attention of the music industry, which had, for the first time, noted a dip global sales (1).

This success raises an important ethical consideration, whether companies could claim that the music they produced was owned in broad terms. In other words, while their property becomes more obvious when it is recorded on a physical CD, and ownership changes with purchase of said CD, could the company claim that the sounds experienced from a digital copy be owned ?

Indeed, could producers claim ownership of a digital copy of the music, and could they claim loss if that copy was shared on the basis of a lost sale *potential* ? This same thought experiment can also extend to other areas of the internet; can digital copies of movies, books, or articles be shared without compensation ? Certainly, the music companies considered that their rights were being abused, and focussed their attentions on Napster (1).

After lengthly legal challenges were lost by Napster, and the creators eventually left the ship, the company sunk to a shadow of its former self. However its concept was tremendously influential, and companies such as iTunes were ready to embrace the digital music market (1), albeit in more legally cautious way.

1. Delbert, C., 2020, Internet Severice Providers, *Major milestones from internet history*, viewed 18/09/2021, <https://www.isp.com/blog/major-milestones-from-internet-history/>
2. Lamont, T., 2013, The Guardian, *Napster: the day the music was set free*, viewed 18/09/2021, <https://www.theguardian.com/music/2013/feb/24/napster-music-free-file-sharing>

5) 2005 YouTube

Former Paypal employees Chad Hurley, Steve Chen and Jawed Karim are responsible for creating a platform that had over 1.5 billion active users a month in 2020 (1), named YouTube. This has increased to over 2 billion in 2021 (2), yet the startup was implemented with initial focus as a private platform, to primarily share video only between acquaintances (1), and there was a slow uptake on users utilising the platform (1). However, in a great example of consumer driven growth, the 20th video, an lipsynced Backstreet Boys' song, reportedly attracted the attention of Susan Wojcicki, then Senior Vice President in marketing at Google, who later advocated for the purchase of Youtube (3). And in 2005 the marketing Nike Ad featuring the soccer player Ronaldinho reached 1 million views (4); an enormous amount for a fledgling startup, and one that truly demonstrated the growth potential of YouTube.

By the following year, YouTube was experiencing rapid growth, with over 19 million users by the first semester of the year (1) and it held over 65% market share of the internet's viewed audiovisual content (1). Even so, it remained free, and like other media sharing endeavours in the past, it was facing copyright lawsuits, which meant that it was financially untenable (1). Yet its potential for business and marketing was recognised by Google, and it was acquired for \$US1.65 billion dollars in October 2006 (1).

With Google's clout, YouTube was propelled to greater and greater levels. Today, 99% of users who access online videos, will use YouTube (1), and containing that level of market share created an unmatched money making opportunity, for users who can monetise their content, and for YouTube who can sell to advertisers or subscribers (5). The company is now worth over \$US100 billion (1) and is invested heavily in its revenue-earning techniques, with a business strategy to keep users watching video on their platform for as long as possible (2).

In effect, YouTube is competing in what is known as the "Attention Economy" (6) where social media companies design their platforms to capture as much of the available time of its users, in order to advertise

to them, and increase their revenue. As such, YouTube introduced a "deep learning" algorithm in 2016 which tracks user usage and promotes additional, personalised content (2) that it believes will most attract the user. The aim is to keep users engaged with the platform for longer, enabling them to increase subscriptions of paid users, or increase revenue from personalised advertisement to its free users. As a reviewer of the recent Netflix documentary "The Social Dilemma" points out, "If you're not paying for the product, you are the product!" (7) which is an interesting and important ethical consideration in the internet today, particularly in the realm of social media competition.

Even so, these ethical consideration should be balanced against the fact that YouTube does embrace the philosophy of information sharing, and holds an enormous amount of educative, informative and entertaining content for ready access.

1. Pereira, M., 2020, Hotmart, *The evolution of YouTube: how did it all start?*, viewed 18/09/2021, <https://blog.hotmart.com/en/history-of-youtube/>
2. Mullery, S., 2021, Tinuiti, *How the YouTube Algorithm Works in 2021*, viewed 18/09/2021, <https://tinuiti.com/blog/paid-social/youtube-algorithm/>
3. 2017, Vox, Full Transcript, *YouTube CEO Susan Wojcicki on Recode Decode*, viewed 18/09/2021, <https://www.vox.com/2017/10/27/16560868/transcript-youtube-ceo-susan-wojcicki-video-recode-decode>
4. Lucas, K., 2021, Givemesport, *Ronaldinho's Nike advert doing the crossbar challenge - was it fake?*, viewed 18/09/2021, <https://www.givemesport.com/1677222-ronaldinhos-nike-advert-doing-the-crossbar-challenge-was-it-fake>
5. YouTube, *How does YouTube make money?*, viewed 18/09/2021, https://www.youtube.com/intl/ALL_au/howyoutubeworks/our-commitments/sharing-revenue/#:~:text=YouTube's main source of revenue,Chat%2C channel memberships and merchandise.
6. Joy, A., 2021, Business Today, *The Attention Economy: Where the Customer Becomes the Product*, viewed 18/09/2021, <https://journal.businesstoday.org/bt-online/2021/the-attention-economy-asher-joy>
7. Hovermann, D., 2020, Medium, *If You Are Not Paying for the Product, You Are the Product!*, viewed 18/09/2021, <https://medium.com/change-your-mind/if-you-are-not-paying-for-the-product-you-are-the-product-4dbc15b9a3f2>

Q1 Define the following technologies

1) Packets

Data travels across the internet in units called packets, which are routed from the origin address to the destination address (1) Fundamentally, as packets form the units of data being transferred, it is a significant part of the technology where information sharing occurs.

Packets are thus the smaller parts of a larger set of information being transferred. While IP packets can potentially be much larger (relatively speaking) a typical packet size is up to around 1,500 bytes (1)(2). This is comprised of a packet header (critical information about the packet, such as source and destination addresses) and the actual data (4), also known as the payload (3).

While theoretically a computer could send a large file directly to another computer in a single long line of unbroken bits, it is not practical to complete this same process to multiple computers simultaneously (3) - these would have to wait until the transfer was completed to all other computers in this theoretical queue. Instead, the internet can utilise the technology of packets in a "packet-switching" fashion, where the packets of data are able to be transferred across various networks, allowing a much more efficient transfer of data (5).

In this fashion, the use of packets and packet-switching technology has contributed to more efficient data flow across the internet.

1. Zola, A., Gillis, A., 2021, Tech Target, *Network Packet*, viewed 12/09/2021, <https://www.techtarget.com/searchnetworking/definition/packet>
2. 2021, Live Action, *What is a Network Packet?*, viewed 12/09/2021, <https://www.liveaction.com/resources/blog/network-packet/>
3. Cloudflare, *What is a packet?*, viewed 12/09/2021, <https://www.cloudflare.com/en-au/learning/network-layer/what-is-a-packet/>
4. Fox, P., Khan Academy, *IP Packets*, viewed 12/09/2021, <https://www.khanacademy.org/computing/computers-and-internet/xcae6f4a7ff015e7d:the-internet/xcae6f4a7ff015e7d:routing-with-redundancy/a/ip-packets>
5. AVI Networks, *Packet Switching*, viewed 12/09/2021, <https://avinetworks.com/glossary/packet-switching/>

2) IP Addresses (IPv4 and IPv6)

IP addresses are essentially unique numbered identifiers for a device on local networks or on the internet (1). The IP represents "Internet Protocol" and lays out the rules, regulations and standards for connecting to the internet, and for routing data, which can allow for bi-directional data flow (2). Given this ability it is an essential technology for the web.

Version 4 (IPv4) addresses are divided into four groups of numbers, separated by a full stop. Each group of numbers ranges from 0-255 because each group of numbers is underlaid by an 8-bit number. Thus, the whole address is a unique 32-bit number (3) which allows unique numbers from 0.0.0.0 to 255.255.255.255. As the number of connected devices in the world reach ever higher numbers, the 4.2 Billion + addresses was an essential foresight in the design.

Version 6 (IPv6) was created in 1998 with the most significant difference to IPv4 in that it is created from 128 bits, instead of 32 (3). This creates an enormous amount more unique addresses to potentially be used, however as the two IP address versions do not communicate with each other directly (2) this creates a difficulty with moving to this technology en masse. Still, the potential for the increased number of unique IP addresses is one solution to the ever-expanding number of connected devices.

1. Kaspersky, 2021, *What is an IP Address - Definition and Explanation*, viewed 12/09/2021, <https://www.kaspersky.com/resource-center/definitions/what-is-an-ip-address>
2. Mozilla, Mozilla VPN, *What is an IP address?*, viewed 12/09/2021, <https://www.mozilla.org/en-US/products/vpn/more/what-is-an-ip-address/>
3. Sebesta, R. (2013) *Programming the World Wide Web*, 7th ed. Boston, Person Education, Inc, p4

3) Routers & Routing

As information moves across the internet in a packet-switching fashion (1), each packet makes a stop at its nearest router, which is a device used in networks that assists packet transfers (2). This device can be physical or virtual (3).

Once a packet is received at a router, the router examines the packet header for its critical information, primarily the destination address (2). Here, the router needs to send the package to the next closest router to the packet's final destination. That router, in turn, does the same, until finally one of the routers in the network is able to forward the packet to its destination IP address. The message travels, or is routed along, a network of routers to reach its final IP address.

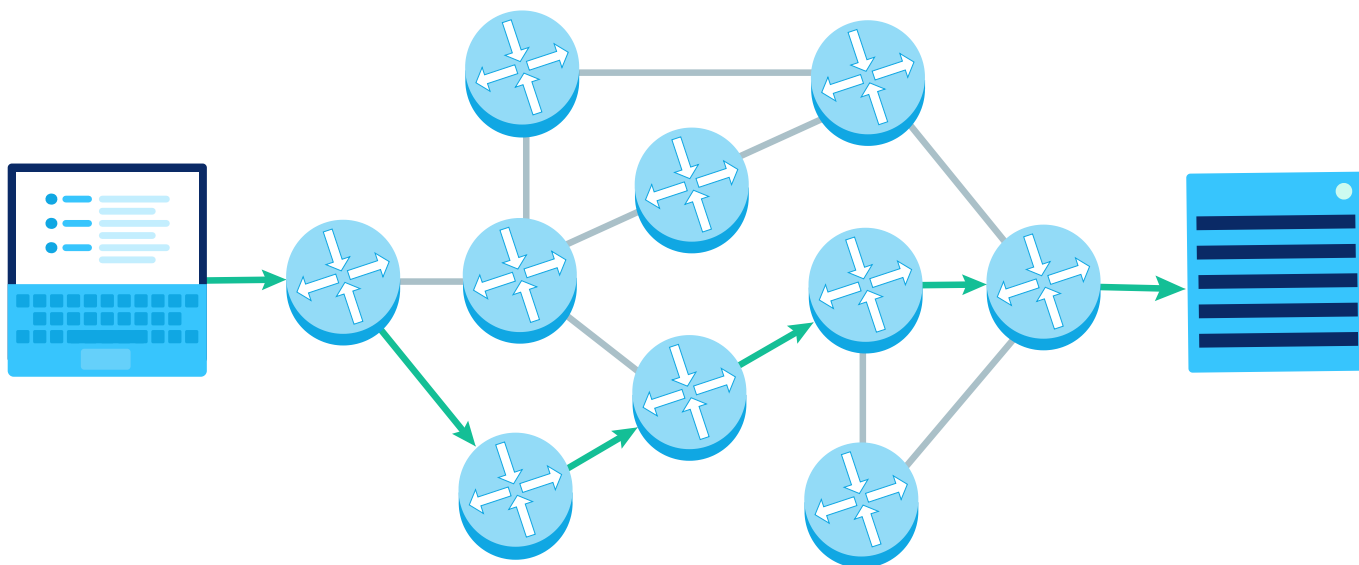


Image source (2)

In this way, routers (and routing) is a critical part of the infrastructure of the internet, creating a network of transfers through a map of routers, so that packets of information can arrive at their destination. Routers are able to guide and direct data through the network, choosing the optimal routes for transferring the packets (4).

One might compare the analogy of a boat carrying cargo (data payload) from the home port (source IP address) to the destination port (destination IP address), noting that it will require stops along the journey from port-to-port (routers), closer and closer to the final address, until it finally drops the cargo off.

This infrastructure's importance cannot be under-stated. Without routers and routing, there would not be the self-same ability to communicate or collaborate over the internet (4) as we have had in the past, and do today.

1. AVI Networks, *Packet Switching*, viewed 12/09/2021, <https://avinetworks.com/glossary/packet-switching/>
2. Fox, P., Khan Academy, *Internet routing protocol*, viewed 12/09/2021, <https://www.khanacademy.org/computing/computers-and-internet/xcae6f4a7ff015e7d:the-internet/xcae6f4a7ff015e7d:routing-with-redundancy/a/internet-routing>

3. Irei, A., 2019, Tech Target, *router*, viewed 12/09/2021, <https://www.techtarget.com/searchnetworking/definition/router>
4. Cisco, What is a router? Key Q&A, viewed 12/09/2021, https://www.cisco.com/c/en_au/solutions/small-business/resource-center/networking/what-is-a-router.html

4) Domains & DNS

The Domain Name System (DNS) is a fundamental part of the internet that connects a directory of names with numbers; these numbers primarily being the IP addresses (1). As it is acknowledged that people have difficulty remembering numbers, internet addresses were given textual names (2) that allowed users to access the addresses with words instead of difficult-to-remember IP addresses. For instance, one could use the IP address 35.246.6.109 or simply use icecream.com.au

As for this example, the DNS icecream.com.au can hold multiple IP records for the same domain name, allowing browsers to attempt connection with multiple IP addresses one-by-one until it receives an appropriate response (3). DNS has thus enabled a single website address (icecream.com.au) to be accessed via multiple IP addresses, which may be more closely routed in various countries of interest.

To delve further into this, domains allow addresses to carry certain characteristics of identity, where fully qualified domain name comprises the hostname and all domain names; the latter of which related to a larger enclosing collection of machines that share the same group identity (2). For instance, using the DNS the educational institutions can belong to the same domain name .edu or .gov which allows ready identity and recognition to users or to groups that might allocate special privileges to those institutions.

For interest purposes, in the 1970's and early 80's, the assignment of names to addresses was completed by one person! Her name was Elizabeth Feinler, who kept a master-list of every connected computer (1), until such time as it became too unwieldy!

1. Shaw, K., Fruhlinger J., 2020, Network World, *What is DNS and how does it work?*, viewed 12/09/2021, <https://www.networkworld.com/article/3268449/what-is-dns-and-how-does-it-work.html>
2. Sebesta, R. (2013) *Programming the World Wide Web*, 7th ed. Boston, Person Education, Inc, p4
3. Tezer, O., 2014, Digital Ocean, *How to configure DNS Round-Robin Load-Balancing For High-Availability*, viewed 12/09/2021, <https://www.digitalocean.com/community/tutorials/how-to-configure-dns-round-robin-load-balancing-for-high-availability>

Q3 Define the following technologies and explain how the technology has contributed to client/server communication development

1) TCP

With multiple computer manufacturers and configurations, a standard was required to allow this range of technology to communicate. This standard was TCP/IP (Transmission Control Protocol / Internet Protocol) (1) and remains one of the most frequently used protocols for end-to-end data transfers (2).

More specifically, the TCP handles packet ordering and checks for errors in the transfer (3); that is, it maintains a connection with the sender to monitor the transfer of data packets, assembling the data on the receiver side, and requesting any missing pieces of data (3), until the transfer of that data is completed. In this way, it can ensure the integrity of data from source to destination (2) which is essential in the information sharing internet where reliable data transfer is required. It acknowledges the unpredictable nature of network behaviors, where packets of data can be disorganised or lost, and TCP is thus able to reorder data or issue requests for re-delivery (4) to the receiver.

While some alternative protocols exist, these may make sacrifices to certain TCP characteristics, such as seen in the User Datagram Protocol (UDP). While UDP is a faster protocol, it lacks certain controls that the TCP handles, such as acknowledgements and connection startups which, while slower, ensure more reliable data transmission (2). So alternative protocols such as UDP may be suited for transmissions such as video streaming, where speed is prioritized over accuracy.

Conversely, high-level protocols prefer the reliability of TCP, such as :

File Transfer Protocol (FTP) (peer-to-peer file sharing)
Secure Shell (SSH) and
Telnet

Source (2)

TCP is also used for email transfers through Internet Message Access Protocol (IMAP) and for accessing the web via HTTP (2) demonstrating the importance in a variety of internet data transfers relying on TCP reliability characteristics.

1. Sebesta, R. (2013) Programming the World Wide Web, 7th ed. Boston, Person Education, Inc, p3
2. Fortinet, *What is TCP?*, viewed 17/09/2021, <https://www.fortinet.com/resources/cyberglossary/tcp-ip?>
3. Cloudflare, *What are IP & TCP?*, viewed 17/09/2021 <https://www.cloudflare.com/en-au/learning/ddos/glossary/tcp-ip/>
4. ExtraHop, *Transmission Control Protocol (TCP)*, viewed 17/09/2021, <https://www.extrahop.com/resources/protocols/tcp/>

2) HTTP and HTTPS

Hypertext Transfer Protocol (HTTP) is a protocol that allows data to be communicated through the World Wide Web (1) by encoding and transporting data between a client and web server (2). Having been invented along with HTML to create the interactive web browser (1), HTTP thus belongs not only in the foundation of data exchanges on the web (3) but remains the primary protocol for data sharing over the internet (2). Users accessing the web will rely on HTTP or HTTPS (explained later) to access website data through their browsers, or send information to servers, as explained below.

Importantly, HTTP itself relies on an underlying protocol such as TCP (2) to form communication connections between clients and servers (1) by managing the reliable transfer of data packets. This allows HTTP to focus not on the transfer of data per se, but rather on how that data is read and processed (4).

The communication method that HTTP relies on is through requests and responses between clients and servers (5), with the requests being initiated through the client recipient, often being the web browser (3), and responses being sent by the server (3). Amongst a host of others, the two most common HTTP

requests are "POST" and "GET" (4) with the first typically indicating that information is being submitted to the server, such as form information (6). The "GET" request seeks information to be returned from the server, such as the details of a website (6) through the HTML document, the scripts and the CSS file (3).

In further detail, the HTTP request is sent with a header section which contains core information such as the method (eg "GET"), the browser being used and the type of data being requested (6). It also contains a body, which carries additional information, including any data being submitted (eg via the method "POST") (6). Responses are typically sent with an HTTP Status code (that indicates whether the HTTP request is successful or not through the use of three-digit codes) (6), their own Response Header and possibly a HTTP Response Body, particularly the case in a successful "GET" method request (6).

It should be noted that HTTP does have security concerns, and can be exploited by malicious actors, who may utilise techniques such as cross-site scripting (XSS) or HTTP request smuggling (1). While not addressing all of the security vulnerabilities of HTTP (1), the use of HTTPS (Hypertext Transfer Protocol Secure) is encrypted to improve the protection of sensitive data being transferred (7) which can become particularly relevant when using the "POST" method to send information such as passwords or credit card information. This encryption security is known as Transport Layer Security (TLS) and was formally known as Secure Sockets Layer (SSL) (7)(a) and makes use of a private key (held by the server) and a public key (available to users) with the former used to decrypt the latter (7).

HTTPS has a further advantage of preventing third parties from injecting unapproved content into their webpages, which can protect against not only malicious actors, but also against Internet Service Providers from injecting their own advertisement content into pages (7)

(a) TLS has been colloquially referred to as SSL, yet SSL is actually deprecated (or supposed to be) due to exploitable vulnerabilities (8)

1. ExtraHop, *Hypertext Transfer Protocol (HTTP)*, viewed 17/09/2021, <https://www.extrahop.com/resources/protocols/http/>
2. NGINX, *What Is HTTP?*, viewed 17/09/2021, <https://www.nginx.com/resources/glossary/http/>
3. MDN Web Docs, *An overview of HTTP*, viewed 17/09/2021, <https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview>
4. Talaber, Z., 2018, TCP vs. HTTP, *What's the Difference, and What's the Big Deal?*, viewed 17/09/2021, <<https://www.extrahop.com/company/blog/2018/tcp-vs-http-differences-explained/>>
5. W3 Schools, *What is HTTP?*, viewed 17/09/2021, https://www.w3schools.com/whatis/whatis_http.asp
6. Cloudflare, *What is HTTP?*, viewed 17/09/2021, <https://www.cloudflare.com/en-au/learning/ddos/glossary/hypertext-transfer-protocol-http/>
7. Cloudflare, *#What is HTTPS?*, viewed 17/09/2021, <https://www.cloudflare.com/en-gb/learning/ssl/what-is-https/>
8. Olenski, J., 2020, GlobalSign *SSL vs TLS - What's the Difference?*, viewed 17/09/2021, <https://www.globalsign.com/en/blog/ssl-vs-tls-difference>

3) Web Browsers (Requests, Rendering & Developer Tools)

This question specifically focuses on components of Web Browsers under specific categories, which can be addressed as follows:

i)

Web Requests through browsers communicate messages to the server, perhaps with a "GET" or "POST" method, as explained in Q3.2. As the browser is the user platform for experiencing the web, it is suitable that requests come directly from the browser, rather than the user having to utilise an additional resource to complete requests. The browser is also able to provide feedback in the format of errors, such as the HTTP 404 "Page Not Found" message (1).

While broadly covered in Q3.2 it is worth examining some of the other web requests that browsers can implement :

GET and **POST** have been explained (Q3.2), where GET has a client expectation of a response from the webserver, while POST can send information from the client to the Web Server, such as form information.

PUT allows the target URL to be replaced or overwritten with a new resource from the client (1)

PATCH modifies a specific part of the resource, so is useful for updating or modifying the resource (1)

DELETE is a request to delete a resource from the server (1)

HEAD request seeks a resource without the body content (2)

TRACE request is to display changes to a resource (2)

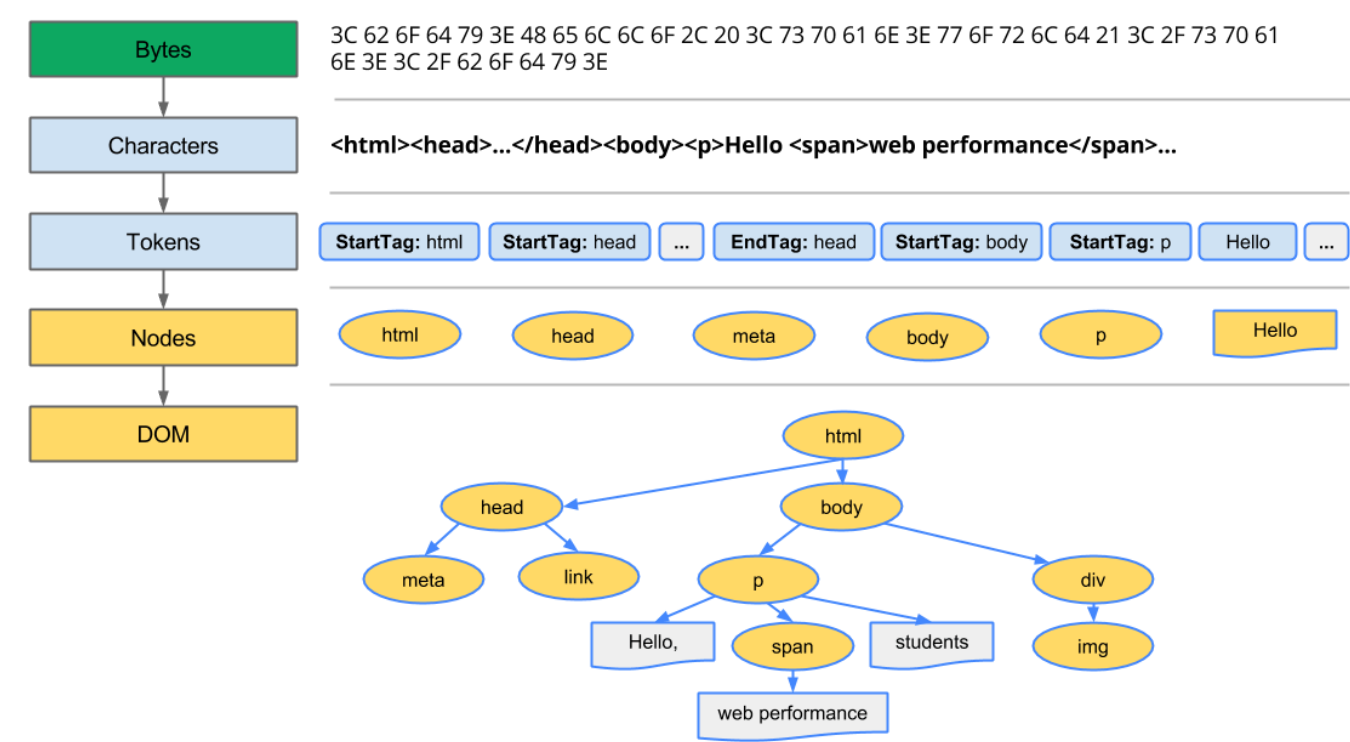
OPTIONS request displays the HTTP methods available for a particular URL (2)

CONNECT request converts the connection to a transparent TCP/IP tunnel (2).

This is relevant when two-way communications are desired to access websites that use HTTPS (3) As before, the browser having the capacity to send these requests when required allows an ease-of-use for the typical web user.

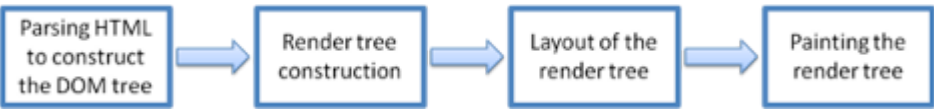
ii)

Web Browser rendering is the process where a browser is able to convert website code into interactive pages when a user visits a particular website (4). This relates to the previously discussed topics, where the browser requests the resource from the server and is able to display, in this case, in appropriate rendered format.



Source (5)

The rendering occurs by capturing the HTML code and constructing a hierarchical Document Object Model representation (a DOM tree if you like) by parsing through the HTML (6) and then representing it into a visual interpretation of the code, drawing in the CSS attributes to display the styling (6). This allows the user to experience information in a structured way with visual impact, rather than simply seeing lines of text.



Source (6)

For further interest, JavaScript is useful for creating an intuitive user experience, but it can be difficult for many search engine bots to process the Javascript, and web crawlers may pass by websites with a heavy reliance on JavaScript for page rendering (4). A process called Dynamic Rendering helps avoid this issue by rendering two types of pages; a static one for the search engine or web crawler, and a more intuitive one for human users (4) to interact with via their browsers.

It can be noted that points (i) and (ii) are stages that occur when a user requires access to a webpage through their browser's ability to request and render it.

iii)
Modern web browsers have access to powerful developer tools, allowing access to a range of assessments, including inspecting HTML, CSS and JavaScript in currently-loaded webpages (7). These allow developers and other users with an interest to view the code and structure of webpages, and how they relate from code to display. It can also be a useful feature to assist with debugging (8) and check the responsiveness of websites to different device specs, such as smaller screen sizes.

There are additional features such as monitors for incoming and outgoing HTTP requests, as well as information about enabled AJAX (9) which is the Asynchronous JavaScript and XML, typically allowing

asynchronous data transfers (such as XML or JSON text) behind the scenes (10).

The Developer Tools also allow viewing of resource storing (such as the local storage and session storage) (9) and gives access to a console, which may work as a log or playground for code outputs (9).

While the depth of features in the developer tools is much more extensive, the features already listed illustrate that the developer tools have a lot of scope, and can thus be a ready-made resource for examining code and making short term changes, such as in the process of debugging, or for locking in long-term modifications, such as utilising the developer tools as a Workspace (9).

1. Source Defence, *Web Request*, viewed 17/09/2021, <https://sourcedefense.com/glossary/web-request/>
2. ExtraHop, *Hypertext Transfer Protocol (HTTP)*, viewed 17/09/2021, <https://www.extrahop.com/resources/protocols/http/>
3. MDN Web Docs, *Connect*, viewed 17/09/2021, <https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/CONNECT>
4. Seobility, *Rendering*, viewed 17/09/2021, <https://www.seobility.net/en/wiki/Rendering>
5. Grigorik, I., 2019, *Constructing the Object Model*, viewed 17/09/2021, <https://developers.google.com/web/fundamentals/performance/critical-rendering-path/constructing-the-object-model>
6. 2014, AMSIVE DIGITAL, *What Does It Mean To "Render" a Webpage?*, viewed 17/09/2021, <https://www.amsivedigital.com/insights/performance-creative/rendering-a-webpage-with-google-webmaster-tools/>
7. MDN Web Docs, *What are browser developer tools?*, viewed 17/09/2021, https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_are_browser_developer_tools
8. MDN Web Docs, *Firefox Developer Tools?*, viewed 17/09/2021, <https://developer.mozilla.org/en-US/docs/Tools>
9. Codementor, *Understanding how to use the Browser Developer Tools*, viewed 17/09/2021, <https://www.codementor.io/learn-development/javascript-css-html-tutorial-front-end-development-tools>
10. W3 Schools, *What is AJAX?*, viewed 17/09/2021, https://www.w3schools.com/whatis/whatis_ajax.asp

Q4 Identify THREE data structures in Ruby and explain the reasons for each

Data structures refer to the arrangement of data, where it is organised and accessed in a specific way (1). This allows different structures to have advantages over another type, particularly where specific scenarios dictate a preference for efficient storage and retrieval of data (2). Three main structures in the Ruby programming language will be described:

1) Array :

These collections of data can hold multiple types of data, such as strings, integers, floats, and other data collections, such as hashes or more arrays! (2). Ruby is able to manage the memory by automatically adjusting when the programmer adds or removes data from arrays (1)(2). The syntax of arrays in Ruby is grouping the data with square brackets, where each piece of data, or object, is categorized as an element of the array, and is separated by commas (2).


```
my_example_array = ["apple", "Ronald McDonald", 123, {key1: value1}, [7, 8, 9]]
```

Some of the uses of arrays include the collection of data outputted from a loop, providing a base for increasingly advanced data structures (1), to be iterated or selected for application to a variety of other processes, algorithms or loops. As arrays are understood to be a set data structure, they follow certain rules, such as obedience to indexing, where the first element is at position 0, the second element is at position 1, etc. If a third item is added to the array, it defaults to the end of the array, and would then fall into element position 2.

This somewhat demonstrates that arrays are ordered, and allows the programmer to base their code around this expectation, such as when the extraction or replacement of data is targeted at specific index positions.

2) Hash :

Here, data is grouped into category pairs, namely a "key" that maps to a "value" (1). The syntax is enclosure with curly brackets, with different syntax options for the key/value pairs, although a typical application would be key:value pairs separated by a colon.

```
my_example_hash1 = {key1: value1, key2: value2, key3: value3}
my_example_hash2 = {"key1"=>"string_value", "key2"=>"string_value"}
```

In this way, hashes are indexed by the key, and thus every key/value pair must have a unique key, which could be a string, integer or, preferably, a symbol (1).

The symbols are preferable as keys due to their memory advantage. As immutable objects that act as identifiers, symbols hold only one space in memory per symbol. So a thousand symbols of identity :key1 hold only one place in memory (3). While hashes have only one unique key, there are instances where multiple *different* hashes will have the same key, which is thus only unique to the individual hash, but multiplied in frequency through the collection of hashes.

For example:

```
my_collection_of_hashes_in_an_array = [ {name: name_of_p1, age: age_of_p1}, {name: name_of_p2, age: age_of_p2}, {name: name_of_p3, age: age_of_p3}]
```

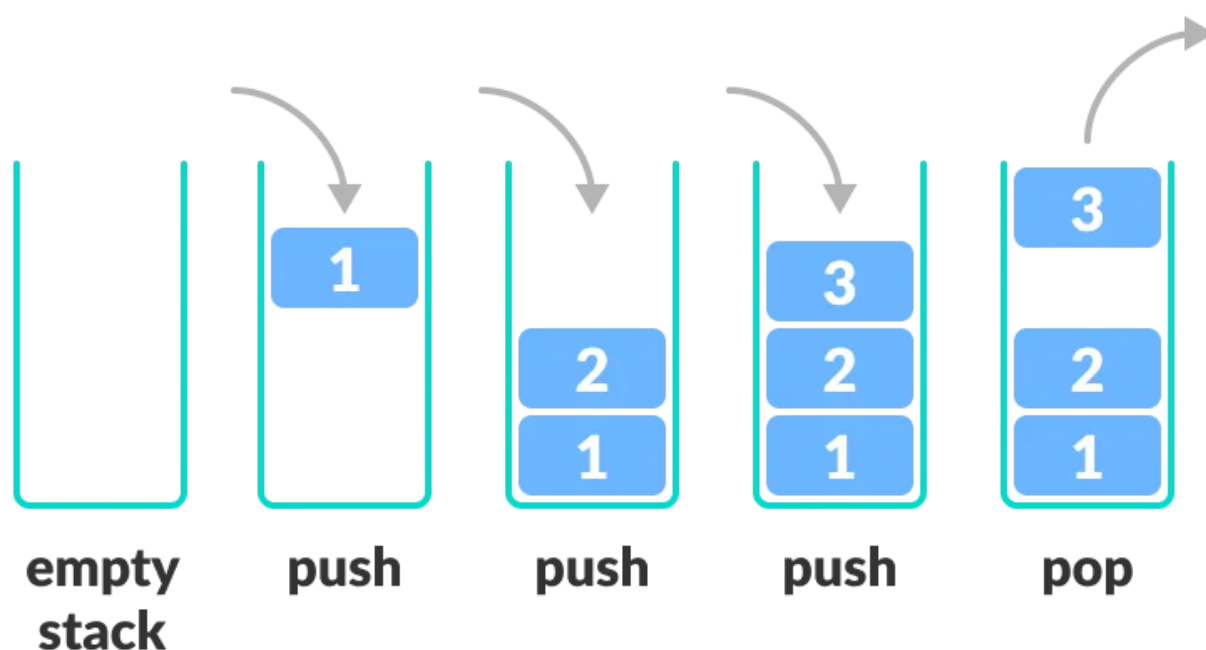
The value, which are paired with keys, can also be of multiple data types, such as strings, integers, floats, arrays and additional hashes (4) They do not need to be unique, even within the same dictionary.

So hashes are advantageous when it is useful for values to be mapped to an identifying key. The example above gives an indication of how a simple database could be built, perhaps with additional characteristics used as keys, like occupation: , hours_worked: , etc. Data can then be drawn out based on the key/value pairs that are of interest.

3) Stack:

The stack follows a linear structure which is Last In First Out (LIFO), also known as First In Last Out (FILO) (5). These data structures can be arranged like a Ruby array, being enclosed in square brackets [1, 2, 3, 4, 5] (2) but is utilised when order and constant time is important (6).

Some memorable examples of where stacks can be used include the "undo" feature in word processors (last change, first one undone) or the back button in browsers (last page visited is first one returned to) (6). Stacks can be used to replace recursive methods with a standard loop, and as seen in the examples prior, maintain an order of items to complete, with the most recent on top (1).



Source (7)

As the image displays, Ruby can utilise the push and pop methods to add and remove data from the stack, similar to an array (8), while maintaining the FILO stack design.

1. Castello, J., Ruby Guides, *An Overview of Data Structures For Ruby Developer*, viewed 17/09/2021, <https://www.rubyguides.com/2019/04/ruby-data-structures/>
2. Shapir, M., 2019, Data Structures in Ruby, *A simple overview of Data Structures with Ruby...*, viewed 17/09/2021, <https://medium.com/@mshapir95/data-structures-in-ruby-a2b709d565be>
3. Criswell, L., 2018, Medium, *Ruby SYmbols vs. Strings*, viewed 17/09/2021, <https://medium.com/@lcriswell/ruby-symbols-vs-strings-248842529fd9>
4. Pearman, N., 2018, Medium, *Introduction to Hashes in Ruby, and in Rails*, viewed 17/09/2021, <https://medium.com/epfl-extension-school/introduction-to-hashes-in-ruby-and-in-rails-be8ac5d4f58a>
5. 2021, Geeks for Geeks, *Stack Data Structure (Introduction and Program)*, viewed 17/09/2021, <https://www.geeksforgeeks.org/stack-data-structure-introduction-program/>
6. Lepore A., 2017, CodeX, *Data Structures in Ruby: Stack*, viewed 17/09/2021, <https://medium.com/codex/data-structures-in-ruby-stack-a83a10a219f1>

7. Programiz, *Stack Data Structure*, viewed 17/09/2021, <https://www.programiz.com/dsa/stack>
8. Castello, J., *Ruby Guides, How to Use Stacks in Ruby to Solve Problems*, viewed 17/09/2021, <https://www.rubyguides.com/2017/03/computer-science-in-ruby-stacks/>

Q5 Describe the features of *interpreters* and *compilers* and how they are different.

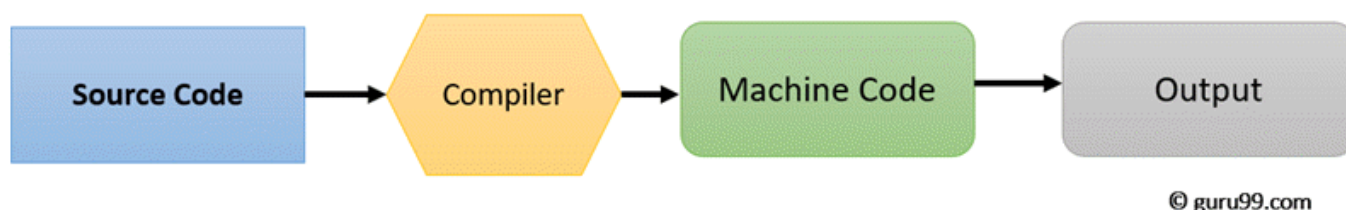
Interpreters and compilers are programs that both help convert high-level languages (the "Source Code") into a language recognised by computers (such as "Machine Code") (1)(2). While high-level languages are considered "human readable" the machine code is made up of binary 1 and 0 bits (2) so requires that conversion.

While both types of programs will complete the conversion, there are some fundamental differences.

Interpreters will translate code "on-the-fly" which means that it translates code line-by-line as the program is running (3). While the interpreter takes very little time to analyze a single line-of-code each time, overall interpreted code runs slower than Compiled code (3). Errors are also revealed one-by-one per line, as this is the method that the Interpreter translates (3)

Compiled code conversely converts all the source code, including pre-compiled code and scripts, into machine code *before* the program is run. While taking time to analyze the entire code beforehand, the compiled code runs faster than interpreted code (2). Compiles will display all errors when going through its process, as it has analyzed the entire program, and might not even complete compilation if the code has significant enough mistakes (3)

How Compiler Works



How Interpreter Works



Image source (2)

Some additional features relating to their differences in executing code include the fact that as Interpreters do not generate Object Code they are memory efficient, whereas Compilers do generate Object Code and require more memory (4)

Compilers allow source code to be private (5) as the conversion occurs into binary code can occur before it is distributed, although it suffers from limited cross-platform support, such as Mac-Windows (5) because they won't utilise the exact same machine code. Interpreters do not have the same privacy, as the source code is largely intact at distribution, and relies on the user machine to have the Interpreter installed. (5)

Languages that *typically* uses Compilers include C, C++ (5) and Java (4) while those that use Interpreters *typically* include Ruby, Javascript and Python (2) although Python and Java may be considered under a *Hybrid* category, along with C# (5) which allow for some variations to the full-compiler method.

While a little outside the scope of this discussion, it worth mentioning that Hybrid languages attempt to gain the a balance of benefits of both Interpreters and Compilers. This is done by compiling the source code to an Intermediate stage, which is as close to machine code as possible while allowing it to be portable and gain cross-platform support (5). It will undergo a final compilation at the user machine with a "Just in Time" compilation (5).

1. Bureau, B., 2019, Business Insider (India), *Difference between Compiler and Interpreter*, viewed 12/09/2021, <https://www.businessinsider.in/difference-between-compiler-and-interpreter/articleshow/69523408.cms>
2. Smith, J., 2021, Guru99, *Compiler vs Interpreter: Complete Difference Between Compiler and Interpreter*, viewed 12/09/2021, <https://www.guru99.com/difference-compiler-vs-interpreter.html>
3. Sassi, R., Better Programming, *Compiler vs. Interpreter: Know The Difference And When To Use Each Of Them*, viewed 12/09/2021, <https://betterprogramming.pub/compiler-vs-interpreter-d0a12ca1c1b6>
4. Programmiz, *Interpreter Vs Compiler: Differences Between Interpreter and Compiler*, viewed 12/09/2012, <https://www.programiz.com/article/difference-compiler-interpreter>
5. Tandiono, J., 2020, *Understanding Compiled | Interpreted | Hybrid languages*, viewed 12/09/2021, <https://jeffrytandiono.medium.com/understanding-compiled-interpreted-hybrid-languages-9764f641faa7>

Q6 Describe two commonly used programming languages and explain the benefits and drawbacks of each

Two commonly used languages are Python and JavaScript. They actually have a lot of in common, and so can be used for similar designs, however specific differences will create advantages for one over the other, which by default creates a possible drawback of sticking with the other. Where possible, flexibility in choosing a language to suit a task is a significant advantage, even with programming languages sharing similarities.

For instance, both Python and Javascript are object oriented (1), and both can be used for scripting purposes (2), however JavaScript is considered the industry standard Scripting Language which allows automated website and app processes (2). Both languages are interpretive languages, although Python additionally offers hybrid capacity, where it can be compiled into a state close to machine code, but still retain cross-platform capabilities, before undergoing final compilation on user machines (3) (see, Q5).

Javascript is much better for mobile development than Python (4) and has significant implementation in over 80% of websites (5) where it provides dynamic functionality on browsers, and runs comfortably on servers (1). JavaScript has the ability to develop both back-end and front-end of applications (6), while

Python is generally more geared towards server-side or backend programming (1). JavaScript is considered the better choice for most websites (7)

The built-ins are different between the two languages, with Python having a significant standard library for data analytics, machine learning and scientific computing (1) whereas JavaScript lacks these specific categories of default library support (1) although the libraries and extensions can be expanded (4). Furthermore, Python has built dictionary capability, as opposed to JavaScript (1), hinting at the fact that Python is useful for large data structure interaction. It can also define data more broadly with mutable and immutable characteristics, and has integer recognition (1) which again hints at its design geared towards data.

Python is considered an easier-to-learn language due to its human-readable syntax, relying on indentation (1) to form blocks of code, and generally requiring fewer lines of code to achieve the same goal (5), adding to the attractiveness of the language. However, JavaScript is considered flexible and scalable enough to be utilised by beginners through to experienced developers (2), so it beginner access is a feature of both languages.

Further discussion on the object orientated nature is worth a mention here; where both languages possess this characteristic, and both possess the ability for procedure programming (9). However Python has several aspects for procedural programming (10) whereas JavaScript does not (4).

In more general comparisons, Python is versatile, open source, has a large number of libraries and a supportive community, and can be a very productive code to work with (8). However, it is not native to a mobile environment, and requires more effort to implement it as such (8). It also has a high memory consumption (8) which may be unfavourable. When used as an interpretive language, it can be slower than some other languages when running (8). It is the principally preferred language for Machine Learning, and is able to deal with complex mathematics and large sets of data, and is unlikely to be superseded in the near future (10).

Again, in general considerations, JavaScript is versatile and fast, as evidenced when run from the client-side, without needing to connect to the server (11). It can be embedded into webpages or inside the scripts of other languages, and allows client-side data validation for faster updates of the browser, given that only selected parts of the page need to be updated, thanks to JavaScript's positioning (11). JavaScript holds a host of advantages over other languages within the web environment (9) however some of the same advantages can be flipped to disadvantages, such as the ease of viewing and implementing code, which creates a mechanism for malicious attacks (11). Also, while browsers can be quite forgiving, JavaScript still relies on their compatibility (11) so the need for integration with browsers is another advantage potentially flipped for JavaScript.

1. Pedamkar, P., *Python vs JavaScript*, viewed 17/09/2021, <https://www.educba.com/python-vs-javascript/>
2. Morris, S., *Tech 101: Python Vs JavaScript-What's The Difference?*, viewed 17/09/2021, <https://skillcrush.com/blog/python-vs-javascript/>
3. Tandiono, J., 2020, *Understanding Compiled | Interpreted | Hybrid languages*, viewed 17/09/2021, <https://jeffrytandiono.medium.com/understanding-compiled-interpreted-hybrid-languages-9764f641faa7>
4. Singh, V., 2021, *Python vs JavaScript: Most Important Differences*, viewed 17/09/2021, <https://hackr.io/blog/python-vs-javascript>

5. Code Institute, *A Comparison of Popular Programming Languages*, viewed 17/09/2021, <https://codeinstitute.net/blog/a-comparison-of-popular-programming-languages/>
6. 2021, Navone, E., *Python VS JavaScript – What are the Key Differences Between The Two Popular Programming Languages?*, viewed 17/09/2021, <https://www.freecodecamp.org/news/python-vs-javascript-what-are-the-key-differences-between-the-two-popular-programming-languages/>
7. Johansson, A., IEEE Computer Society, *5 Reasons JavaScript Is Still Better Than Python*, viewed 17/09/2021, <https://www.computer.org/publications/tech-news/build-your-career/5-reasons-javascript-is-still-better-than-python>
8. Krzysztof, B., 2018, Netguru, *Python Pros and Cons*, viewed 17/09/2021, <https://www.netguru.com/blog/python-pros-and-cons>
9. MDN Web Docs, *About JavaScript What is JavaScript?*, viewed 17/09/2021, https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript
10. Wilfred, T., 2021, Codersera, *Python Vs Javascript: A Detailed Comparison*, viewed 17/09/2021, <https://codersera.com/blog/python-vs-javascript/>
11. Data Flair, *Pros and Cons of JavaScript – Weigh them and Choose wisely!*, viewed 17/09/2021, <https://data-flair.training/blogs/advantages-disadvantages-javascript/>

Q7 Describe two ethical issues (from the nominated areas) and discuss the extent to which an IT professional is ethically responsible.

Ethical issues are interesting and important, and this question deserves and exploration of the issues it presents, and a conversational or exploratory response has been chosen to answer this question. Throughout this discussion, it will be evidenced that when ethics are considered, the guidelines may prove to be incomplete, inconsistent, and thus of limited help to the IT Professional.

i) Privacy laws are of paramount consideration in the tech industry currently, particularly in today's world, where there are competing rights to privacy versus health, crime, ease-of-access, and profit motivations. This conversation will focus on the crime potential versus right of privacy, and examine some of the difficulties that may exist when an IT professional believes that ethics do not align with the current laws, or where they are compelled to act against the interests of what they consider to be ethically acceptable products, such as encrypted products.

To delve into this further, the example of The Assistance and Access Act 2018 can be examined. Here, it is alleged that over 95% of ASIO's (the Australian Security Intelligence Organisation) targeted terrorist individuals use encrypted communications, and 90% of priority cases use encryption technology (1). As such, the Australian Government issued broader powers to the law enforcement and intelligence agencies under this Telecommunications and Other Legislation Amendment (Assistance and Access) Act 2018, with the endeavour to being able to compel access to encrypted communications (2).

The difficulties with these implementations is multifold, including the communication of such laws.

Firstly, Home Affairs writes that nothing in the laws can require the industry to break encryption (1) and the Government cannot build a decryption, interception or data retention capability, or implement backdoors, or make less secure an innocent person's communication (1). Firstly, set aside that this last statement may in fact contradict the presumption of innocence, when these laws are implemented in an investigation prior any potential conviction.

Secondly, further contradictions arise when more closely examined, whereupon the Act is amended to provide a "framework for voluntary and mandatory industry assistance" (2), which entail; i) Technical Assistance Requests, which are voluntary, requiring the use of decryption or other data access capacity (2) ii) Technical Capability Notices, which are compulsory, and compel access to data that is currently accessible (2) iii) Technical Capability Notices, which are compulsory, and compel the provider to build new capabilities for access (2)

So, immediately, confusion may be created as to what these laws can actually compel the IT Professional, and puts at risk a lack of compliance by individuals and companies, particularly when these requests or notices can be directed to anyone who "develops, supplies or updates software used" (2).

Furthermore, representatives of the Five Eyes alliance (a security alliance between seven countries including Australia) released a statement in October 2020 advising that while they "support strong encryption" (3) that tech companies are asked to build backdoors to access content (4) which either shows a misunderstanding of the encryption system, or is a misrepresentation of their stance. As former chief security officer Alex Stamos puts it, backdoors into encryption is like "drilling a hole in the windshield" and compromising the entire structure, or compromising the whole encryption integrity (4).

It is noted that there is no "golden key" concept, allowing access into that company's software, that doesn't also carry the potential to be used by malicious actors (5), and the situation is such that recommendations on the ethical steps asks that other ethical considerations are set aside or neglected.

So the Australian laws and surrounding guidance may be considered unhelpful in assisting the developer to act in an ethical way, and indeed, may even be unhelpful in directing the developer to act in a lawful way, when seemingly contradictory statements are given. It is this author's opinion that such laws thus require the assistance of experienced legal professionals to offer guidance, rather than relying on the broad and potentially contradictory definitions provided by Government, or attempting the difficult interpretation of Act Amendments oneself.

The best an IT Professional can do is to work within the legal framework to best act ethically, while considering the balance of other ethical duties that they are committed to. And so attention can now be directed towards the ethical balance, where a developer may ethically believe in the presumption of innocence, and thus decline to decrease the encryption security, if able to, of a person under investigation, but not yet convicted of a crime. This law has the potential to challenge that ethic, rather than offering guidance for that developer to act in an ethical way. Indeed, in the USA, the EARN IT Act created a "best practices" checklist for companies to follow to best avoid criminal and civil liability, and yet could be determined to include weakened encryption as a requirement (5), indicating a guideline that drifts away from the ethic of maintaining privacy that the developer, or his/her company may have.

It may also be the desire of the developer to provide a product to their customers that guarantee's the safety and privacy of their data, perhaps in the use of zero-knowledge technology, which is where service providers do not know anything about the data that is stored on their servers (6). The ethics of this service relate to a foundational belief in the security and privacy of that service, for that particular user. Thus, a conflict of interest and ethics is created if or when a Government compels access to that information.

Now, while appearing less sinister when being compelled by the Australian Government (with a perceived corruption score of 77/100, with 0 being highly corrupt, and 100 being very clean) (7) versus a Government in Somalia (perceived corruption score of 12/100) (7) it nonetheless should be objectively considered, "Is it

ethically ok to guarantee privacy to any person, and then abandon that guarantee, at the behest of any Government?"

1. 2019, Department of Home Affairs, *The Assistance and Access Act 2018*, viewed 18/09/2021, <https://www.homeaffairs.gov.au/about-us/our-portfolios/national-security/lawful-access-telecommunications/data-encryption>
2. Stilgherrian, 2021, Carnegie Endowment for International Peace, *The Encryption Debate in Australia: 2021 Update*, viewed 18/09/2021, <https://carnegieendowment.org/2021/03/31/encryption-debate-in-australia-2021-update-pub-84237>
3. 2020, United States Department of Justice, *International Statement: End-To-End Encryption and Public Safety*, viewed 18/09/2021, <https://www.justice.gov/opa/pr/international-statement-end-end-encryption-and-public-safety>
4. Ray, S., 2020, Forbes, *United States, Six Other Nations Ask Tech Companies To Build Backdoors To Encrypted Communications*, viewed 18/09/2021, <https://www.forbes.com/sites/siladityaray/2020/10/12/united-states-six-other-nations-ask-tech-companies-to-build-backdoors-to-encrypted-communications/?sh=75cc81574051>
5. Laliberte, M., 2020, Help Net Security, *Why building backdoors into encryption won't make us safer*, viewed 18/08/2021.
6. Lam, I, 2016, Tresorit, *What is Zero-Knowledge Encryption?*, viewed 18/09/2021, <https://tresorit.com/blog/zero-knowledge-encryption/>
7. Transparency International, *Corruption Perceptions Index*, viewed 18/09/2021, <https://www.transparency.org/en/cpi/2020/index/nzl>

ii) Another difficult area of ethics and appropriate guidance is freedom of thought, conscience, speech and the media, where this discussion will focus on freedom of speech. Importantly, there is no express Constitutional protection of freedom of speech in federal legislation in Australia (1) compared with the United States, which hold a First Amendment right guaranteeing that freedom of speech (2). This may be relevant given the size of the influences of the United States technology industry, the access of that industry to Australian citizen developers, and the current discussion of social media's selective platforming (3). It may also be worth some discussion from an American position, given the sheer size of their industry, and the headquartering of the largest Social Media platforms in the USA (4).

So this question can be approached in a few ways, with consideration towards the ethical rights that a developer has to their own expression, and the responsibilities towards upholding other persons freedom of speech. The first is largely the same as persons of any occupation, so the focus will be directed to the latter.

Assume that a developer created a platform for discussion, where users could sign-in and leave comments. The ethical considerations that the IT Professional have is whether they have an active duty to monitor and remove comments, and to what level this decision is made. For instance, should they worry if potentially defamatory comments are made by an anonymous user ? Should they remove comments that exhibit discrimination, and does it matter who the discrimination is directed towards ? If they disagree with a comment, should they remove the comment simply because it disagrees with their personal opinion, and thus tilt the majority of comments to support their own views ? Or are they ethically required to leave it ?

These questions, and many more, fall to the IT professional who is either moderating the platform, or working for a company whose policies make that direction. While Australia does not have a Constitutionally

endorsed right to freedom of speech (1) it does have some laws that give guidance to whether certain speech is legally appropriate or not.

For instance, Section 18C of the Racial Discrimination Act applies to speech on ethnic origin, race and skin colour, where it is unlawful to "offend, insult, humiliate or intimidate" a person on the basis of their race (5). This was brought under the spotlight in 2011 when articles written by Andrew Bolt were published and found to contravene Section 18C in the *Eatock v. Bolt* (2011) case (6) which demonstrated that even seasoned writers could be held accountable. While the notions of "offend" and "insult" may vary widely in the nature of subjective interpretation, courts have found that conduct must cause "profound and serious" consequences, and not "mere slights" (7), which, while still subjective, give some balance back to the freedom of speech. Section 18D further provides an exemption to Section 18C where it is an "expression of genuine belief held by the person making the comment" or "done reasonably and in good faith" (6).

On this basis, it would be ethically acceptable to allow comments on race that are done in good faith, held by genuine belief, even if they could cause insult. If the exemptions do not apply, and it is extremely difficult to know in many cases if they would, then speech falling under Section 18C is not only unethical, but also unlawful.

Another principle of law relates to defamatory speech, where such publications are made on Social Media, and are legally liable for civil defamation claims. For instance, in 2013 the payment of over \$100,000 was ordered for damages against a music teacher, by another party who posted defamatory tweets and Facebook posts (8). Australian Law allows for claims of defamation without first establishing any minimum level of harm, and a default presumption of reputation damage is supported (8).

As such, the IT Professional can understand that certain defamatory posts made on their platform, are not only unethical, but may be the catalyst to a civil suit.

So, ethically it now must be considered whether the IT professional is responsible for the comments made by someone else, particularly when commented on the platform under the purview of the IT professional. A recent decision by the High Court in Australia would suggest so (9) where it ruled that comments or posts from the general public are the legal liability of media companies if those comments are made on the company's Facebook page (9). It is thus considered the ethical responsibility of these companies, or the IT Professionals that work for them, to remove comments that could be liable for defamation.

This may have broader implications for other companies, not just media organisations. Brett Walker, an expert in communications law, described how businesses, sporting clubs and community groups could fall under the same liability (9), giving strong indications on the ethical responsibility of IT Professionals to remove such liable comments.

It is even suggestive that IT Professionals should be ethically compelled to advise companies of this potential liability before establishing comment pages that are linked to that company. That way, the company only gives informed consent to the IT Professional, now being fully aware of the risks that this High Court decision reveals.

In March 2021, Facebook made an announcement that comments could now be controlled by the user for their page, and Twitter enabled greater control over who could reply to tweets on a user's account (10) perhaps in relation to the original decisions of the NSW Supreme Court in 2019 (10). Indeed, some companies, such as the independent SE Voice newspaper, will choose to err on the side of heavy

moderating, limiting the scope of freedom of speech, but protecting their potential liability more cautiously (9).

This Australian case draws some parallels to an argument in the United States, where social media companies are protected under Section 230 of the Communications Decency Act (11). This is because they are considered as platforms that are largely considered not responsible for the content that the general public post (11) similar to the way that a Telecommunications company is not held liable for the content of a phone conversation that their users have. In contrast, editorial publishers exhibit active moderating, and thus decide on the content being published, and so can be held responsible for slanderous or defamatory articles (11).

The conflict in this definition arises when it is noted that when social media sites such as Facebook and Twitter do censor content, and remove user access, they are exhibiting editorial power (12), and not merely acting as an open platform. For instance, in the lead up to the United States presidential elections, Twitter and Facebook both took steps to limit access to a New York Post story that was disparaging towards the son of a presidential candidate (13). Rightly or wrongly, this exhibited a tilt towards the domain of a publisher.

So the ethics do change, as do the ethical responsibilities of the IT Professional, in relationship to the role they hold.

If the social media company is a publisher, then the IT Professional remains ethically responsible to protect against litigation by not publishing liable content. They also remain ethically responsible to ensure that content aligns with their company ethos, unless within a capacity reserved for open discussion. If they are a platform, then the ethical responsibility lies in promoting open discussion at all times, even if comments are oppositional to their own viewpoint, or the viewpoint of their company. Just like the telephone company should keep the lines open, even if they theoretically might disagree with the content of the conversations. In this latter case, the IT Professional is only ethically motivated to censor on the basis of speech that falls outside of the First Amendment, or outside of Australian laws, such as Section 18C/18D of the Racial Discrimination Act or falls into the parameters of defamation.

As Australia's technology sector grows, it will be important to understand that the guidance given to IT professionals to act ethically will need to be weighed up against a range of other measures, including potentially contradictory ethical alignments, such as freedoms to privacy and speech. The ethical responsibilities will also be better clarified if the structures of the companies the IT Professionals are working in are more clearly defined, so that the roles are equally well understood, and their associated ethical bearings revealed.

1. Australian Human Rights Commission, *Freedom of expression and the Internet*, viewed 18/09/2021, <https://humanrights.gov.au/our-work/3-freedom-expression-and-internet#fn19>
2. Cornell Law School, *First Amendment*, viewed 18/09/2021, https://www.law.cornell.edu/constitution/first_amendment#:~:text=The First Amendment guarantees freedoms,and the right to petition.&text=It guarantees freedom of expression,of individuals to speak freely.
3. Petricone, M., 2021, Morning Consult, *The Facts on Social Media Bias*, viewed 18/09/2021, <https://morningconsult.com/opinions/the-facts-on-social-media-bias/>
4. Golden, *List of social media companies*, viewed 18/08/2021, <https://golden.com/list-of-social-media-companies>

5. Australian Human Rights Commission, *At a glance: Racial vilification under sections 18C and 18D of the Racial Discrimination Act 1975 (Cth)*, viewed 18/09/2021, <https://humanrights.gov.au/our-work/race-discrimination/projects/glance-racial-vilification-under-sections-18c-and-18d-racial>
6. Global Freedom of Expression, *Eatock v. Bolt*, viewed 18/09/2021, <https://globalfreedomofexpression.columbia.edu/cases/eatock-v-bolt/>
7. Australian Human Rights Commission, *The AHRC and the Racial Discrimination Act: setting the record straight*, viewed 18/09/2021, <https://humanrights.gov.au/about/news/opinions/ahrc-and-racial-discrimination-act-setting-record-straight>
8. Rolph, D., 2016, *The Conversation*, *Social media and defamation law pose threats to free speech, and it's time for reform*, viewed 18/09/2021, <https://theconversation.com/social-media-and-defamation-law-pose-threats-to-free-speech-and-its-time-for-reform-64864>
9. Doran, M., 2021, ABC News, *Facebook defamation ruling by High Court exposes all page owners to lawsuits, not just the media*, viewed 18/09/2021, <https://www.abc.net.au/news/2021-09-12/facebook-defamation-high-court-ruling-exposes-more-than-media/100451198>
10. Taylor, J., 2021, The Guardian, *Facebook now lets users and pages turn off comments on their posts*, viewed 18/09/2021, <https://www.theguardian.com/media/2021/mar/31/facebook-turn-off-comments-on-post-limit-restrict-disable-comment-posts-moderation-control-tool>
11. Zilles, C., 2020, Social Media HQ, *If Social Media Companies Are Publishers and Not Platforms, That Changes Everything*, viewed 18/09/2021, <https://socialmediahq.com/if-social-media-companies-are-publishers-and-not-platforms-that-changes-everything/>
12. Schillings, 2021, Lexology, *Social media companies are acting like publishers, so they should expect to be regulated like one*, viewed 18/09/2021, <https://www.lexology.com/library/detail.aspx?g=4fec37ff-cf69-4cb1-af64-0040d57b1135>
13. Fox, C., 2020, BBC News, *Twitter and Facebook's action over Joe Biden article reignites bias claims*, viewed 18/09/2021, <https://www.bbc.com/news/technology-54552101>

Q8 Explain control flow, using an example from Ruby programming language

In programming terms, control flow refers to the order that statements in a code are executed (1) and evaluated (2). *Broadly speaking* this is from the first line of code to the last line of code (1). This description is clarified as "broadly speaking" because there are frequent structures implemented within code that modify this control flow, such as conditionals, loops (1) and calling on functions. So it might be considered as the way that the code is executed based on control structures that that self-same code contains.

Using a couple of examples in Ruby, it can be demonstrated as follows:

```
x = 7
puts x + y
y = 4
```

In this case, the statements are executed line by line. 7 is stored in variable x. The result of variable x (7) + y (not defined) is requested. This returns an error ("undefined local variable or method...")

A human may be able to glance ahead, and understand that y should represent 4, and establish that x + y is thus equal to 11. However, the computer obeys control flow guidance, and will not "glance" ahead (at least,

not without specific instructions to do so, which is the critical point here).

```
x = 7
y = 4
puts x + y
```

This now follows in-built control flow (first line to last) and will output 11.

It is worth noting the "control" part of control flow. Here, the programmer is able to direct the computer to follow a specific pathway, that is not simply first-line to last-line.

```
x = 1
while x < 5
  puts "x is currently equal to #{x} which is less than the value of 5"
  puts "x can now be increased by 1"
  x += 1
end
x = 100
```

Here the code begins by reading first-line to next consecutive line, however the programmer has utilised techniques to create a desired control flow. In this case, by using a "while loop". There are 7 lines of code. Rather than executing 1-7 this program will run lines 3-5 as long as the condition on line 2 is "true".

As before, the program does not "glance ahead" to see that eventually x is to be equal to 100. Instead the programmer controls the flow to stay within the while loop, until created conditions (the value of x in relation to 5) are satisfied, in which case the program will continue in a line-by-line fashion, moving from line 2 and jumping to line 7.

Several examples of control flow structures that Ruby programming offer include statements such as "Break", "Next", "Redo", "Retry", "Return", "Throw/catch" (3)

1. MDN Web Docs, *Control Flow*, viewed 12/09/2021, https://developer.mozilla.org/en-US/docs/Glossary/Control_flow
2. Computer Hope, 2018, *Control FLOW*, viewed 12/09/2021, <https://www.computerhope.com/jargon/c/contflow.htm>
3. Dumitru, S., 2019, *Control Flow in Ruby*, viewed 12/09/2021, <https://medium.com/@soni.dumitru/control-flow-in-ruby-18bdd6bdb92d>

Q9 Explain type coercion

There are cases where data of one type (such as integers) may be required as another type (such as strings). This is where type coercion comes into play, where there is automatic or implicit (but not *explicit*) conversion of one type to another (1). It can largely be considered where the interpreter or compiler

determines what type will be applied when values are being compared of different types (2). Here, the values may be coerced so that they can better interact.

For instance, this example modified from source (1) is a JavaScript example.

```
const my_string = '4';
const my_integer = 3;
let my_sum = my_string + my_integer;
console.log(my_sum);
```

Here, the output to console will be a string "43". JavaScript automatically coerced the integer (`my_integer`) into a string, so that it could concatenate the two of them.

This differs from *type conversion* which is a broader umbrella term, and may include implicit or explicit conversions (1). For instance, in Ruby, we could explicitly convert our string "5" to an integer (example 1) or vice-versa (example 2)

Example (1)

```
x = "5"
x = x.to_i
x == 5 (true)
```

Example (2)

```
y = 5
y = y.to_s
y == "5" (true)
```

1. MDN Web Docs, *Type coercion*, viewed 12/09/2021, https://developer.mozilla.org/en-US/docs/Glossary/Type_coercion
2. McFarlin, T., 2014, envatotuts+, "The Beginner's Guide to Type Coercion: What is Coercion?", viewed 12/09/2021, <https://code.tutsplus.com/articles/the-beginners-guide-to-type-coercion-what-is-coercion--cms-21917>

Q10 Explain data types, using examples

Data types are an important concept in programming languages, where data can be stored as a variety of types, which interact differently (1) in their environment. Data types can be considered as an attribute that, when linked with the data, guides the computer system how to interpret its value (2).

This acknowledges that different data types can allow different features, for instance integers and floating point numbers can be used for mathematically reasoning, while boolean or bool types can carry the

value of true or false (3) making them ideal for conditional statements (4).

Further to this, different programming languages define data types differently (5) so it can be tricky to make comparisons, although there are some common themes.

For instance, most languages acknowledge the different types of Integer (whole numbers) and Floating Point (fractional numbers in decimal) (5) while some languages use the type "Number" to incorporate both (2).

Some languages recognise Character type (5) which is a single letter or digit, a punctuation mark or symbol, and even a blank space (2) and String type is a popular reference to a sequence of characters (2). In the programming language Python, a single character is referenced as a string with a length of 1 (6) and, like many other programming languages, strings are recognised as arrays of bytes (6).

This leads on to some collective data types, such as Arrays or Lists that have an ordered collection of data contained therein (2). Different languages will have rules as to the types of data types stored within arrays or lists, such as a homogenous array which can only store values of the same type (7). Or a heterogeneous list in Python will usually store homogenous objects, but are easily able to store multiple values of different types, such as strings, integers, other lists and dictionaries (8).

Some final other examples of Data Types include those recognised by programming language Ruby, such as Hashes (values linked to a unique key) and Symbols, with the latter allowing an alternative to naming a variable with a string, and offering the advantage of better memory performance (9). This is linked to their immutable characteristic and ability to be stored in the one memory location, even if the same symbol is used multiple times to hold a value, such as with keys in hashes (10)

From this discussion, it can be noted that many different data types exist, and while programming languages have their individual interpretations, there are common threads between the languages and their types. Overall, they retain a consistent philosophy which is that the type determines how the system will recognise and be able to utilise that particular value of that particular type.

1. W3 Schools, Python Data Types, *Built-in Data Types*, viewed 12/09/2021, https://www.w3schools.com/python/python_datatypes.asp
2. Choudhury, A., *dataled academy, _What are Data Types and Why are They Important*, viewed 12/09/2021, <https://dataled.academy/guides/data-types/>
3. 2021, Computer Hope, *Boolean*, viewed 12/09/2021, <https://www.computerhope.com/jargon/b/boolean.htm>
4. Busbee, K., Braunshweig, D., *Programming Fundamentals, Boolean Data Type*, viewed 12/09/2021, <https://press.rebus.community/programmingfundamentals/chapter/boolean-data-type/#:~:text=A Boolean data type has,in the mid 19th century.>
5. Future Learn, *Using data types and variables*, viewed 12/09/2021, <https://www.futurelearn.com/info/courses/begin-programming/0/steps/2940>
6. W3 Schools, *Python Strings*, viewed 12/09/2021, https://www.w3schools.com/python/python_strings.asp
7. Singh, C., *Beginners Book, Data Structure - Array*, viewed 12/09/2021, <https://beginnersbook.com/2018/10/data-structure-array/#:~:text=An array is a collection,double%2C float%2C char etc.>
8. Python Programming, *Python Lists*, viewed 12/09/2021, <https://www.pythonprogramming.in/list.html#:~:text=List can contain heterogeneous values,in some>

kind of order.

9. Geeks for Geeks, *Ruby / Data Types*, viewed 12/09/2021, <https://www.geeksforgeeks.org/ruby-data-types/>
10. Criswell, L., 2018, Medium, *Ruby Symbols vs. Strings*, viewed 12/09/2021, <https://medium.com/@lcriswell/ruby-symbols-vs-strings-248842529fd9>

Q11 Restaurant Problem. Identify classes to help solve the problem, with a short explanation as to why those classes would be used.

This is an interesting problem, with a variety of different approaches, and appears to be largely exploratory in nature, so this is the approach that will be taken.

The scenario advises that all staff have quit. This could then be assumed to be wait staff, chefs, cleaners, etc. In this scenario, the business then becomes one where human resources are used offsite, such as contracting to another restaurant. The business model then shifts to an order based one, where customers can select their food of their choice, and an order is generated, payment is made, and the food is delivered. Similar models exist outside of this hypothetical, so it seems an appropriate understanding of the problem and solution. We can make our situation a little more unique, where we are responsible for ordering and paying for ingredients and consumables.

Classes allow the blueprints of the models. With this in mind, classes that could help to build the application might include:

class Customer

with class attributes including @@number of customers instance attributes including

```
@customer_name
@customer_address (for delivery)
@customer_payment_details
@possible_password (validation)
@an_order_history, perhaps composed of arrays
```

methods include the ability to add to the instantiated object with user input incrementally (first name / last name / address number / address street / etc)

This could be utilised for ongoing customers, where the details could be pre-filled. Otherwise, it is at least needed for payment and delivery

Without knowing the precise mechanisms, it would also be worth exploring whether a separate payment object is beneficial, rather than storing as a single attribute in the class Customer

Classes allow the blueprints of the models. With this in mind, classes that could help to build the application might include:

class Payment

with class attributes including @@payment_types, perhaps in a hash for extractable data based on type (key) and frequency (value) for follow up reviews instance attributes including

```
@payment_type
@details_of_account (such as credit card number)
@authorisation_received

methods include the ability to add to the instantiated object with user
input incrementally (payment choice / number / expiration / etc)
```

class Order

with class attributes including @@total_number_of_orders, @@total_cost_of_making orders, and @@total_prices_charged, thus allowing profit calculation instance attributes including

```
@food_items chosen for order
>> including item frequency, size and customisation (example, gluten free)

@drink_items chosen for order
>> including item frequency, size and customisation (example, diet)
@is_payment_fulfilled ?
@date / time of order
@possible_discounts / vouchers

methods include the ability to add to or remove from instantiated object
attributes (for instance, add a food item, and a drink, change mind on the
food item). It can also cater for additional conditions to be added, such
as gluten-free.
```

This becomes useful in its capacity to send the order to the restaurant that is making the meal, by exporting the data held in the attributes. It could be stored and/or sent in a number of structures, such as {type: pizza, topping: hawaiian, size: large, number: 2} and when the values are drawn out, the order that the kitchen receives is an easy to read x2 Haiwain Pizza (Large)

class Delivery

class attributes include number of deliveries inside 2km, within 2-5km, within 10km, 15km+ instance attributes could include

```
@delivery_cost
@order_on_way ?
@departure_address
@departure_time
@delivery_address
@delivery_time
@perhaps_draw in a resource to establish the distance travelled for data
use
```

This allows for application to provide live updates, and feedback for the company, which may assist targeting customers in certain areas

class Feedback

class attributes include @@frequency of 5 star reviews, 4, 3, 2 and 1 star will also be accepted instance attributes

```
@optional_name
@return_contact_detail
@rating
@commented_feedback (with character limit!)
```

Here, we can add some more specifics. For instance, if the kitchen was being contracted, but this business owner had to purchase all the consumables

class Consumables

with class attribute including total expense and total income, allowing a simple calculation and storage of profit, and also instance attributes include

```
@name of item
@cost of item to purchase
@cost of item to sell
```

class Meal < Consumables

class attributes including stock levels of consumables, which can be quite extensive will inherit the attributes of its parent class Consumables

```
@inherited cost to buy / cost to sell (totals)
@ingredient1_name / cost / amount required / price to buy
@ingredient2_name / cost / amount required / price to buy
@ingredient3_name / cost / amount required / price to buy
@ingredient4_name / cost / amount required / price to buy
    -etc-

@packaging1_name / order_code / amount used / price to buy
@packaging2_name / order_code / amount used / price to buy
@packaging3_name / order_code / amount used price to buy
    -etc-

@special_consideration1 (example gluten free option)

@size (perhaps default small, medium, large options)
```

```
> method can allow the ingredient list to be true / false, allowing
customisation within the object
> methods include the automated reminder to order more ingredients when
stock low
```

class Drinks < Consumables

class attributes including stock levels of consumables, which can be quite extensive will inherit the attributes of its parent class Consumables

```
@inherited cost to buy / cost to sell (totals)
@ingredient1_name / cost / amount required / price to buy
@ingredient2_name / cost / amount required / price to buy
    -etc-

@packaging1_name / order_code / amount used / price to buy
@packaging2_name / order_code / amount used / price to buy
    -etc-

@special_consideration1 (example diet option)

@size (perhaps default small, medium, large options)

> method can allow the ingredient list to be true / false, allowing
customisation within the object
> methods include the automated reminder to order more ingredients when
stock low
```

These consumables burrow down to the objects that are constructed by the user, and are the product of the company, rather than the business model components. As such, there is some additional customisation, with separate attributes to identify not only ingredient types, but also amounts used.

This set of classes can be quite expandable, and would be useful in the context of a business plan, where the ins-and-outs of the business model are intimately known. Even with initial considerations, it can be seen that the list of main classes being considered is filling out considerably. It must also be designed in such a way that additional modifications can be made, and so classes should be designed in such a way as to allow for changed circumstances, such as decreasing or increasing menu items, or updated financials.

Q12 Identify and Explain the Error in the Code (below) that is preventing correct execution of the program

```
1 celsius = gets
2 fahrenheit = (celsius * 9 / 5) + 32
3 print "The result is: "
4 print fahrenheit
5 puts "."
```

The issue is related to an understanding of types

When utilising the "gets" method in Ruby, the return is a string (1). So the variable, celcius, is storing a string. When performing multiplication, Ruby actually has the ability to join a string by the number of times it is *multiplied* by (or, in this case, concatenated with duplicates of itself).

For instance

```
our_variable = "hello" * 3
p our_variable
"hellohellohello"
```

Once the above question 12 code requests a *division* however, Ruby will not be able to divide the string, running into an error. Looking ahead, a concatenation of the string with an integer would also have been problematic, although our order-of-operations presents the initial problem.

Furthermore, using the "gets" method returns not only a string, but a trailing whitespace `\n`, which should be rememebered. The example above, if using the "gets" method to input hello, without chaining a `".chomp"` method, and then multiplying it, would have resulted in :

```
"hello\nhello\nhello\n"
```

The initial issue here, is perhaps that the user input was never meant to be a string. It was probably meant to be an integer. If it was an integer, then we could apply mathematical operands to it, such as multiplication, division and addition.

A possible fix would be to convert the input into an integer, before it is even stored in the variable, celcius, by chaining the `.to_i` (to integer) method to the "gets" method. While we might consider using a `.chomp`

to remove the trailing whitespace, Ruby has the ability to convert the string into an integer without this method being applied.

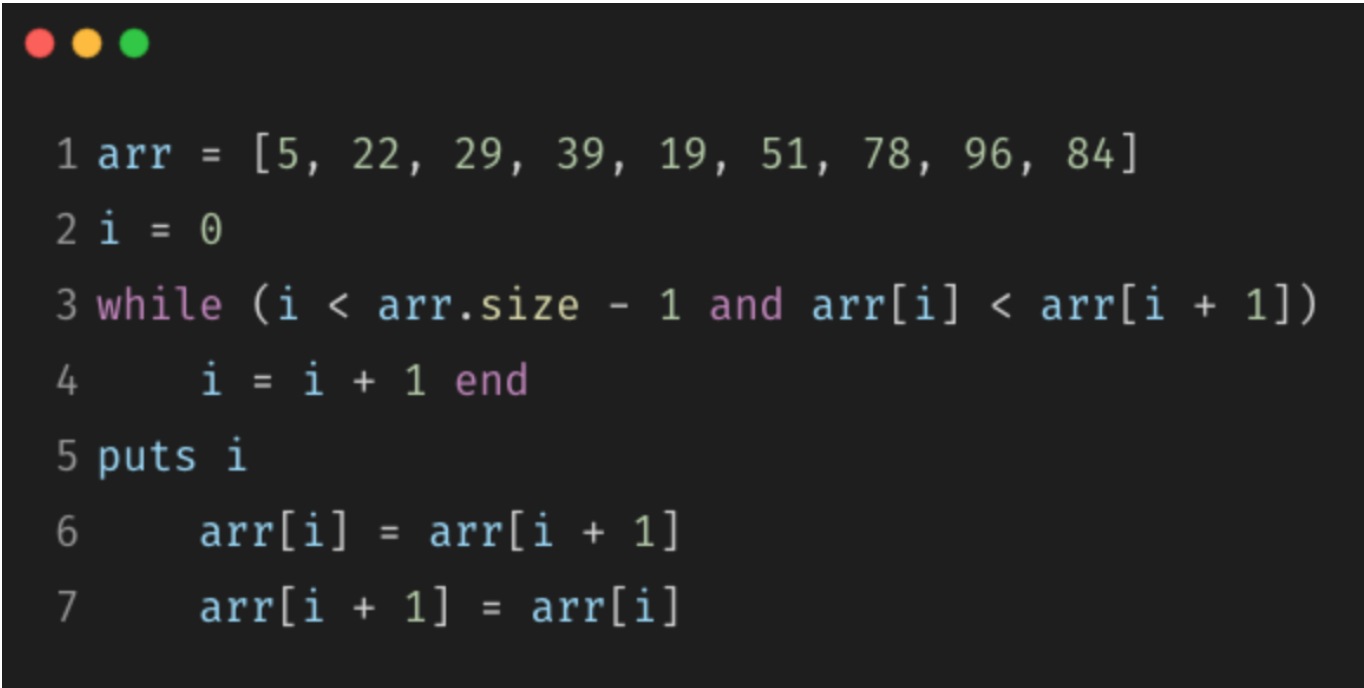
```
celcius = gets.to_i
```

As a point of interest, utilising interpolation could tidy up the final 3 lines of code, while still retaining easy readability.

```
celcius = gets.to_i
fahrenheit = (celcius * 9/5) + 32
puts "The result is: #{fahrenheit}."
```

1. Castello, J., Ruby Guides, *How to Use The Ruby Gets & Ruby Chomp Methods*, viewed 12/09/2021, <https://www.rubyguides.com/2019/10/ruby-chomp-gets/>

Q13 Rewrite the supplied code so that only the first two elements are swapped.



```
1 arr = [5, 22, 29, 39, 19, 51, 78, 96, 84]
2 i = 0
3 while (i < arr.size - 1 and arr[i] < arr[i + 1])
4     i = i + 1 end
5 puts i
6     arr[i] = arr[i + 1]
7     arr[i + 1] = arr[i]
```

This question is requesting that only the first two elements that are out of order are to be swapped, which leaves the assumption that the remaining elements, even if out of order, do not need to be re-arranged.

If this is the case, then the solution is relatively straightforward. Firstly, the section of code that is problematic should be examined:

```
arr[i] = arr[i+1]  
arr[i+1] = arr[i]
```

This will run into issues because it is similar to :

```
x = y  
y = x
```

Were the value of y is already stored in x, the second line is essentially :

```
y = y
```

We need an interim variable to house one of these values if we want to swap them :

```
temp_var = y  
y = x  
x = temp_variable
```

or for our example :

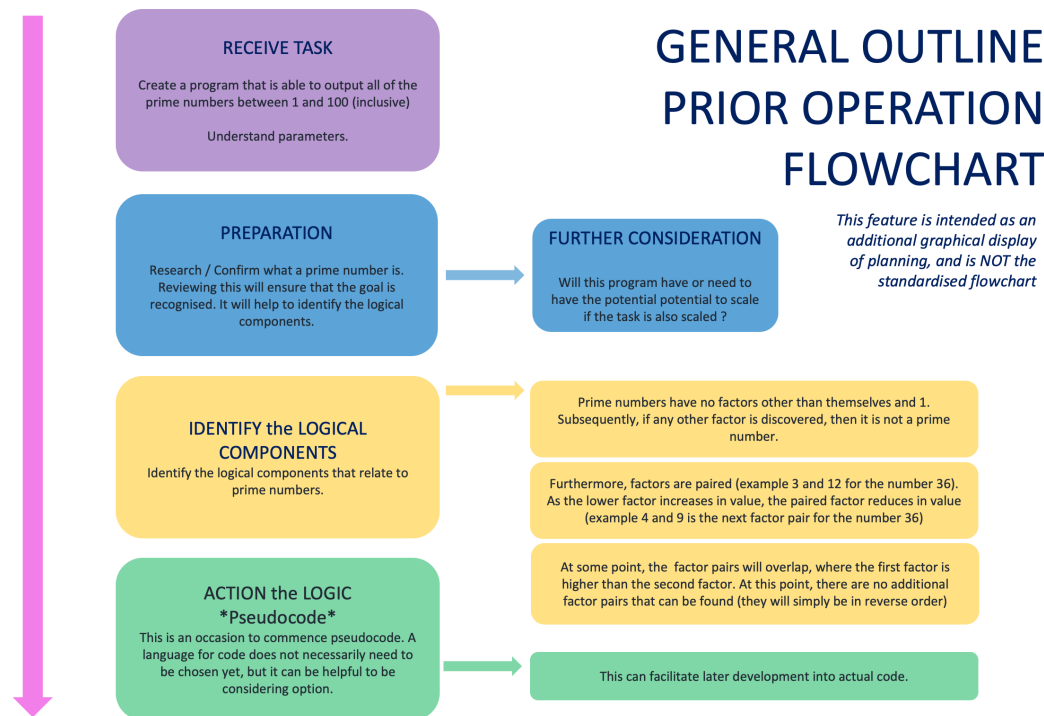
```
> temp_var = arr[i+1]  
> arr[i+1] = arr[i]  
> arr[i] = temp_var
```

If we wanted to continue sorting this array in this fasion, then we would need to continue iterating through our array, swapping additional values under the same conditions. Once the whole array had been iterated through, the process is restarted, again and again, until all values are in their correct order.

While helpful for debugging, the "puts" statement can be removed if not desired.

Q14 Demonstrate algorithmic thinking through completing two tasks

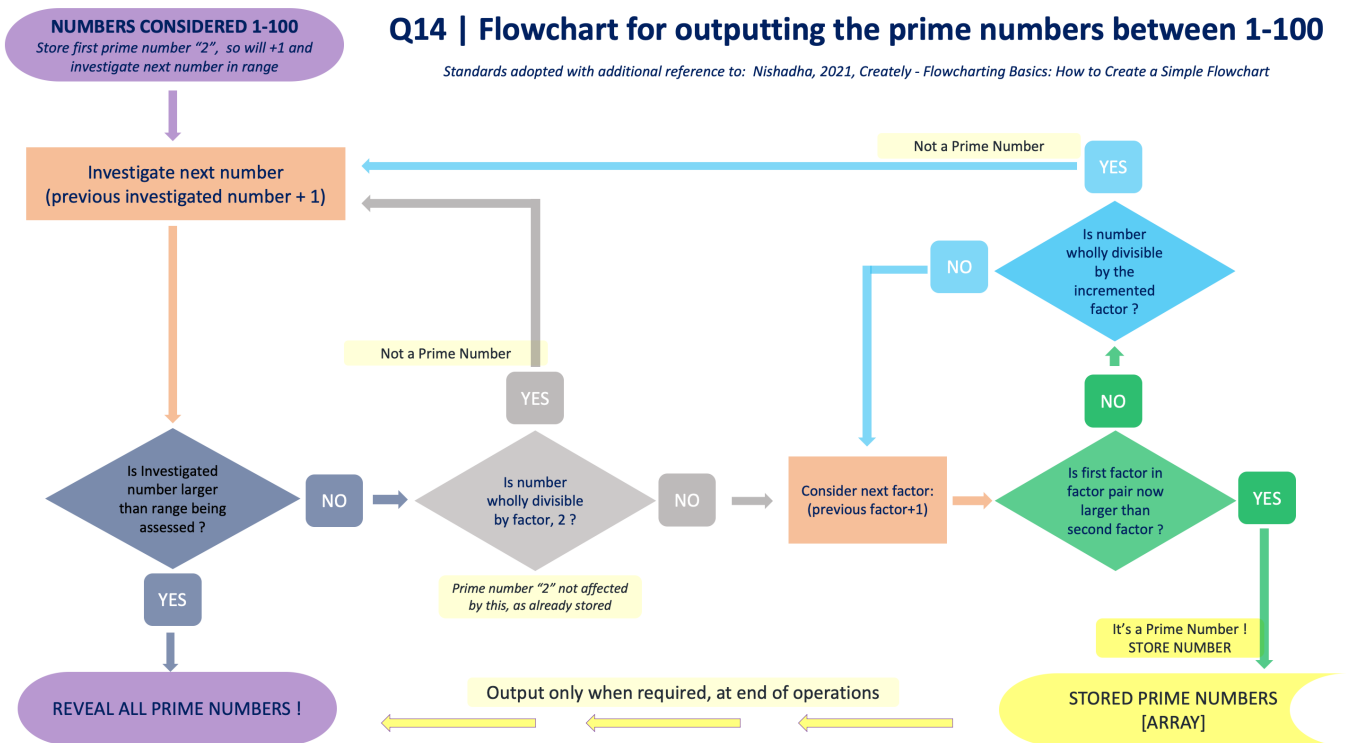
Firstly, a general outline of planning will assist prior creation of our pseudocode and flowchart. Please be aware that this first image is not the standardised flowchart, but rather an additional outline of initial steps in our consideration of the problem.



Create a flowchart outlining steps for calculating and outputting all prime numbers between 1 and 100 (inclusive)

Standard conventions give guidance to the format of the flowchart. In this case, there are multiple resources to use, and a simple guide was sourced:

Nishadha, 2021, Creately - Flowcharting Basics: How to Create a Simple Flowchart,
<https://creately.com/blog/diagrams/all-you-need-to-know-about-flowcharting/>



Write pseudocode for the process outlined in flowchart

1. Numbers to be considered are 1 – 100
2. 1 is not considered a prime number
3. Which makes 2 our first prime number
4. It is a special case of prime number. All other even numbers are not prime numbers.
5. A collection of prime numbers will need to be kept.
6. The first prime number "2" can begin this collection.
7. Therefore our first number to investigate is 3
8. We will iterate through all numbers up to 100
9. If any number that is more than two can be wholly divisible by 2 (an even number) then it is not a prime number.
10. If that is the case, then the next number in the range can be immediately investigated.
11. If the number is not divisible by two, it remains a potential prime number. Is it wholly divisible by the next potential factor, being 3 in this case?

12. If that is the case, then the next number in the range can be immediately investigated
13. Follow this pattern : if the number is not divisible by our potential factor, then check if it is divisible by the next potential factor (previous factor + 1)
14. As all factors are paired, if the lower potential factor ever reaches a point, following this procedure, that it is greater than its pair, then the number under investigation is a prime number.
15. This prime number needs to be added to a collection of other discovered prime numbers
16. Then, once done, the next number in the range can be investigated, following the same process
17. Eventually the range of numbers being investigated will be exhausted, and once this happens, the program can stop assessing numbers.
18. At this stage, all prime numbers can be outputted or displayed to the user, in a choice of formats.

While outside the scope of this task, with this logic in mind, the code can now be created, such as this author's example below.

```
prime_numbers = [2]

# as this problem requests values 0-100, we can start our array of prime
# numbers with 2, the first prime number. This avoids an early conditional
# statement dilemma, which removes even numbers.

# if the range was more diverse, further conditions would be required, or
# potentially the ability to input, or the creation of a method with
# parameters to set the range.

investigated_number = 3

# as per above, given the range parameters, our investigation starts at
# number 3, as 2 is already recorded in the array

possible_factor = 2

# the first possible factor to investigate is 2 (even numbers)

while investigated_number < 101

    if investigated_number % possible_factor == 0
        possible_factor = 2 #resets to first factor
        investigated_number += 1
```

```

    next
  elsif possible_factor > (investigated_number / possible_factor) #all
factors thus checked
    prime_numbers.append(investigated_number)
    possible_factor = 2 #resets to first factor
    investigated_number += 1
  next
  else possible_factor += 1 #checks next factor-pair
end
end

p prime_numbers

#various alternatives for output, such as puts for a lengthy vertical
list, or interpolation for a more structured output.

```

Q15 Write pseudocode OR Ruby code for the problem:

The design of this code does not restrict the user to using only keys that are nominated to be selected, and outputs guidance for when incorrect keystrokes are used. For instance, "Y" or "N" are valid options, but "y" and "n" are accepted, and "x" will return a prompt to select the correct key.

Also, there are safeguards on converting a string from the `gets` method to an integer, where if the user selects a character, "a" and it is converted to an integer, Ruby returns 0, which is considered an integer by the program. The safeguard is one that compares the original value with the converted value, to see if an integer was originally placed.

The use of nested loops encapsulates the answers, meaning that the output relies on the sufficient answering of both questions, "Is it raining?" and "What is the temperature?".

```

# This problem provides two variables, raining and temperature
raining = true
temperature = 0

# Any other variables in this code are not REQUIRED for the code to
function, but rather improve flow control

#There are additional newline characters utilised to grand additional
readability in terminal

adequate_answer = false
while adequate_answer == false

  puts 'Is it raining ? Please only enter Y or N'

  input_raining = gets.chomp.downcase
  case input_raining
  when 'y'
    raining = true

```

```
when 'n'
  raining = false
else
  puts "\n**** Please only enter a Y or N in answer to this question
****\n\n"
  next
end

puts 'What is the temperature ? Please enter a number in digit format
only ? eg 10'

input_temperature = gets.chomp
input_temperature_dance = input_temperature.to_i.to_s

# this sets up variables to compare with each other to see if they are
numbers or not

# gets brings in a string, .to_i method convers to integer, but also
converts words to 0.0

# which will be recognised as an integer. "Word" => 0.0 => "0.0" BUT
"15" => 15 => "15"

if !(input_temperature == input_temperature_dance)
  puts "*** This is not a number, or not a number in the correct format
(digit) ***\n* Please enter a number in digit format only, such as 10.
Please retart input.*\n\n"
  next
else
  adequate_answer = true
  input_temperature = input_temperature.to_i # back to integer. Use this
one to keep readable

  if raining && (input_temperature < 15)
    print "It's wet and cold\n" # print statement requested instead of
puts. Newline desired.

  elsif !raining && (input_temperature < 15)
    print "It's not raining but cold\n"

  elsif !raining && (input_temperature >= 15)
    print "It's warm but not raining\n"

  else
    puts "It's warm and raining" # no print statement requested for this
section, so chose to use a puts statement.
  end
end
puts "\nThank you for using this weather program"
```

Q16 ACME Corporation are hiring a new junior developer, as part of their hiring criteria they've created a "coding skill score"

Here, there are scores attributed to various language competencies. The question outline requests user input, to nominate their various skillset, and provides them with an overall score, as well as indicating the skills they did not select, how much each skill is valued, and how much they would improve their score.

The following code creates some additional interaction, as well as catering for incorrect input selections.

```
#initial template of languages and scores

score_template = [
  { name: 'Python', score: 1, selected: '[ ]' },
  { name: 'Ruby', score: 2, selected: '[ ]' },
  { name: 'Bash', score: 4, selected: '[ ]' },
  { name: 'Git', score: 8, selected: '[ ]' },
  { name: 'HTML', score: 16, selected: '[ ]' },
  { name: 'TDD', score: 32, selected: '[ ]' },
  { name: 'CSS', score: 64, selected: '[ ]' },
  { name: 'JavaScript', score: 128, selected: '[ ]' }
]

# calculation methods and display methods, held in separate functions to
increase encapsulation

def display_score_template(score_template)
  puts "\n"
  score_template.each_with_index do |language, index|
    puts "#{language[:selected]} #{index + 1}. #{language[:name]}"
  end
end

def calc_existing_skill_score(existing_skillset)
  existing_skill_score = 0
  existing_skillset.each do |iteration|
    existing_skill_score += iteration[:score].to_i
  end
  existing_skill_score
end

def nope_msg # this is just a bit of fun :)
  system("clear") || system("cls")

  puts "\n"
  puts '| \ | | ^^^ | | ^^^ | | ^^^ |'
  puts '| \ | |   | |   | |   |'
  puts '| \ | |   | |   | |   |'
  puts "\n\n"
end
```



```

def calc_upskill_scores(available_upskills)
  available_skill_points = 0
  available_upskills.each do |language|
    available_skill_points += language[:score].to_i
  end
  available_skill_points
end

# this summary method allows for other calculations to be drawn in, and
# then the display is formatted for easy user readability

def upskill_points_assessment(available_upskills)
  puts "If you wanted to upskill, you could consider studying these
languages :\n\n"
  puts '=== LANGUAGES'.ljust(23) + 'SKILL SCORE POINTS', '-' * 42

  available_skill_points = calc_upskill_scores(available_upskills)
  available_upskills.each do |language|
    puts "=== #{language[:name]}".ljust(30, '_') +
(language[:score]).to_s.rjust(3, '_')
  end
  puts ('-' * 6).rjust(34), available_skill_points.to_s.rjust(33)
  puts "\n",
    "If you learnt all of the above languages you\ncould increase your
skill score by #{available_skill_points} points !"
  puts "\n\n"
end

##### CODE CODE CODE #####

existing_skillset = []
options_chosen = []
available_upskills = score_template.clone

puts "\nHere is a list of relevant programming languages that ACME
Corporation are interested in\n\n"

display_score_template(score_template)

adequate_answer = false
while !adequate_answer

  puts "\nAre you competent in any of these languages? Please enter Y or
N"
  initial_answer = gets.chomp.downcase # the downcase is to cater for
either upper or lowercase entries

  if initial_answer == 'y'
    adequate_answer = true
    all_done = false
    puts "\nGreat! Which of the languages are you competent in ?\n\n"

    while !all_done

```

```

display_score_template(score_template)
puts "\nPlease select a number of the language you are competent in,
and press enter. Enter X to exit."
input = gets.chomp

if input.downcase == 'x'
  all_done = true

  # iterate through existing skills. If part of the available
  upskill list, remove these items (as already has them)

  existing_skillset.each do |iteration|
    available_upskills.delete(iteration) if
available_upskills.include?(iteration)
  end

  elsif (input.to_i < 1) || (input.to_i > score_template.length)

    # chose this option rather than "8" in case of future adjustment
    with additional number of languages

    system("clear") || system("cls")
    puts "\n\n*** Please only choose a number between 1 and #
{score_template.length} ***\n\n"

    elsif options_chosen.include?(input.to_i)
      system("clear") || system("cls")
      puts "\n\n*** You have already chosen this option !! Pick another
option or enter X to exit ***\n\n"

    else
      system("clear") || system("cls")
      score_template[(input.to_i - 1)][:selected] = '[x]' # checkmarks
the score_template for display
      options_chosen.append(input.to_i) # keeps a list of options
chosen, so duplicates are not an issue
      existing_skillset.append(score_template[(input.to_i - 1)]) #
existing skillset list is appended by choices
    end
  end

  elsif initial_answer == 'n'
    adequate_answer = true

  else
    nope_msg()
    puts 'Please enter only Y or N'

  end
end
end

```

#as the two directions (zero skills or 1 || or skills) will end in the same position, with the finalising of loops and scores being displayed,

the single finish works well at the end of our code, and draws in our final methods to complete.

```
system("clear") || system("cls")
puts "\nThank you for your input. Under ACME Corporation guidelines your
languages skill score is #{calc_existing_skill_score(existing_skillset)}"
upskill_points_assessment(available_upskills)
```

While this code format suits the purpose of the assignment question, and fulfils the requirements, if there were additional needs to scale, or save the results per entry, then the use of a class structure could be implemented, with each instance saved to file. This would further take advantage of the Object Oriented Programming nature of Ruby.

Thank you for reading. T