"The only true wisdom is in knowing you know nothing."

— Socrates

Manjunath.R

#16/1, 8th Main Road, Shivanagar, Rajajinagar, Bangalore560010, Karnataka, India

*Corresponding Author Email: manjunath5496@gmail.com

*Website: http://www.myw3schools.com/
• Educational institutions are teaching it
• Corporate societies are employing it
• Pupils need it
• (Pedagogues desire it... ;)
• (Coders perceive it... :)

- ENIAC - the world’s first completely electronic computer conceived and constructed on Penn’s campus in 1946.
- **Operating system**: A well-defined set of instructions in the form of statements that is installed into the computer which provide instructions for computer how to operate (i.e., how to receive the raw data through input devices (like key board, mouse etc.), process the input data through processing device called CPU (Central Processing Unit) and store the processed data (in information storage devices like hard disks) and display the processed data through output devices (like monitor, printer etc.)). A well-defined instruction is called a code and a well-defined set of instructions constitute a program (i.e., compilation of codes gives a program). For example: word is a code and a paragraph is a program (i.e., compilation of words gives a paragraph).
- IBM 5155 – the early portable computer introduced in February 1984
- **Examples of Operating system** (a well-defined set of instructions that is installed into the computer which provide instructions for computer how to operate) are: DOS (Disk operating system developed by Bill Gates and Paul Allen in 1980 for IBM PCs), Linux (operating system developed by Linus Torvalds at the University of Helsinki with the assistance of developers around the world), Windows NT, 95 & 2000 (developed by Microsoft corporation for PC), UNIX (developed by AT&T Bell Laboratories, Murray Hill, New Jersey) etc.
- **Drivers**: A well-defined set of instructions (what we call programs or software) that is installed into computer and stored in the form of files in the computer that allows the computer to communicate with its hardware components (i.e., hardware components like mouse, key board, printer etc.). Without drivers, the computer cannot communicate with
its hardware components – as a result a mouse, keyboard, or a printer won’t work properly.

- **Hosting:** Host is a system that contains information and this information can be accessed by computer users by a means of internet. This process is called hosting.

- **IP address or Internet Protocol address:** Just like every house on a street has a postal address which helps the post man to find that house on a street, every computer connected to internet has an Internet Protocol address or IP address which helps the other computers to find that computer on the network.

- Suppose A, B, C, D, E, F and G – are the computers connected to each other by means of internet (i.e., they are on the network). If computer A has not assigned any IP address, then users at computers B, C, D, E, F and G cannot send any email or other data to user at computer A or user at computer A cannot receive any email or other data from the users at computers B, C, D, E, F and G by a means of internet.

- **IP address is of four types:**
  - Public IP address and Private IP address
  - Static and dynamic IP address
    - Static IP address → permanent IP address
    - Dynamic IP address → temporary IP address (exist only for a limited time i.e., IP address leased for a limited time).
  - Public and Private IP address:
    
    Amazon organization is assigned an IP address IPₐ and Google organization is assigned an IP address IP₉. And the systems (1, 2, 3, 4, 5…..etc.) within the Amazon or the Google organization are assigned an IP addresses IP₁, IP₂, IP₃ … etc.

    IPₐ and IP₉ imply public IP addresses
    IP₁, IP₂, IP₃ … etc. implies private IP addresses

    Which means: Public IP address is used for external communication (i.e., used for the communication between the Amazon and the Google organization) and Private IP address is used for internal communication (i.e., used for communication between the systems within the Amazon or the Google organization).

- **Domain name:** If we type www.google.com (which is called the domain name) in the browser, then the domain name is sent to DNS (domain name system) where the domain
www.google.com is converted to IP address 74.125.224.72 (because website / web pages are only identified by their IP address in the server) and this IP Address is sent to the web server (a system that acts like a data center from where the required information (i.e., web page of google.com) is taken and sent to the browser and the www.google.com web page is displayed in the web browser). If you type the IP address in the browser, then DNS is not required. For human convenience (difficult remember numbers, for example: www.google.com is domain name, IP address is 74.125.224.72. Because it is difficult to remember 74.125.224.72 so www.google.com is preferred).

- **ASP.NET:**
  - ASP → Active Server Page
  - ASP.NET (Active Server Page Network Enabled Technology) is a technology developed by Microsoft corporation using the languages -- C#, Visual Basic. Net, J script & J# -- to build dynamic web pages / websites and web applications.

- Dynamic web page contains information (say date, month or year or time zone of the day) change automatically daily without a developer editing its source codes while static web page contains information (say date, month or year or time zone of the day) cannot change automatically daily without a developer editing its source codes.

- **Virtual Memory:** If the RAM (i.e., Random Access Memory) is full and it is running out of space available for storage of further information and there is no access to store further information, the idea of extending memory by using disk is called virtual memory (i.e., the further information is stored in disk and retrieved when required). This process is called paging or swapping.
What is Linux and why is it so popular?

Whether you know it or not you are already using Linux (the best-known and most-used open source operating system) every day. From supercomputers to smartphones, the Linux operating system is everywhere. As an operating system, Linux is a family of open source Unix-like software based on the Linux kernel - that sits underneath all of the other software on a computer, receiving requests from those programs and relaying these requests to the computer's hardware. With regard to careers, it is becoming increasingly valuable to have Linux skills rather than just knowing how to use Windows. In general, Linux is harder to manage than Windows, but offers more flexibility and configuration options.

Every desktop computer uses an operating system. The most popular operating systems in use today are: Windows, Mac OS, and LINUX. Linux is the best-known notoriously reliable and highly secure open source portable operating system -- very much like UNIX -- that has become very popular over the last several years -- created as a task done for pleasure by Linus Torvalds -- computer science student at the University of Helsinki in Finland -- in the early 1990s and later developed by more than a thousand people around the world.

Linux is fast, free and easy to use, that sits underneath all the other software on a computer -- runs your computer -- handling all interactions between you and the hardware i.e., whether you're typing a letter, calculating a money budget, or managing your food recipes on your computer, the Linux operating system (similar to other Operating Systems, such as Windows XP, Windows 7, Windows 8, and Mac OS X) provides the essential air that your computer breathes.

Linux is the most important technology advancement of the twenty-first century and Licensed under the General Public License (GPL) that Linux uses ensures that the software will always be open to anyone and whose source code is open and available for any user to check, which makes it easier to find and repair vulnerabilities and it power the laptops, development machines and
servers at Google, Facebook, Twitter, NASA, and New York Stock Exchange, just to name a few. Linux has many more features to amaze its users such as: Live CD/USB, Graphical user interface (X Window System) etc.

**Why LINUX?**

Although Microsoft Windows (which is the most likely the victim of viruses and malware) has made great improvements in reliability in recent years, it considered less reliable than Linux. Linux is notoriously reliable and secure and it is free from constant battling viruses and malware (which may affect your desktops, laptops, and servers by corrupting files, causing slow downs, crashes, costly repairs and taking over basic functions of your operating system) – and it keep yourself free from licensing fees i.e., zero cost of entry ... as in free. You can install Linux on as many reliable computer ecosystems on the planet as you like without paying a cent for software or server licensing. While Microsoft Windows usually costs between $99.00 and $199.00 USD for each licensed copy and fear of losing data.

Below are some examples of where Linux is being used today:

- Android phones and tablets
- Servers
- TV, Cameras, DVD players, etc.
- Amazon
- Google
- U.S. Postal service
- New York Stock Exchange

Linux Operating System has primarily three components:
• **Kernel**

Kernel is the core part of Linux Operating System and interacts directly with hardware. It is responsible for all major activities of the Linux operating system.

• **System Library**

System libraries are special programs using which application programs accesses Kernel's features.

• **System Utility**

System Utility programs are responsible to do specialized tasks.

**Important features of Linux Operating System:**

- Portable
- Open Source
- Multi-User
- Multiprogramming
- Hierarchical File System
- Security

Now Linux (successfully being used by several millions of users worldwide) has grown passed the stage where it was almost exclusively an academic system, useful only to a handful of people with a technical background. It provides more than the operating system: there is an entire infrastructure supporting the chain of effort of creating an operating system, of making and testing programs for it, of bringing everything to the users, of supplying maintenance, updates and support and customizations, runs on different platforms including the Intel and Alpha platform. Today, Linux is ready to accept the challenge of a fast-changing world to do various
types of operations, call application programs etc. Since the hiring focus is shifting more and more toward DevOps type skills, a Linux skill set will be the types of things that will make you very deployable.

The command-line interface is one of the nearly all well built trademarks of Linux. There exists an ocean of Linux commands, permitting you to do nearly everything you can be under the impression of doing on your Linux operating system. Although, this to the end of time creates a problem: by all of so copious commands accessible to manage, you don't comprehend where and at which point to fly learning them, especially when you are learner. If you are facing this problem, and are peering for a painless method to begin your command line journey in Linux, you've come to the right place, we will launch you to a hold of well liked and helpful Linux commands.

**Description:**

Display system date and time.

**Command:**

date

**Description:**

Display calendar.

**Command:**

cal
Description:
Display date, time and calendar.

Command:
date & cal

Description:
Display August month 2016 year calendar.

Command:
cal 8 2016

Description:
Used to clear the terminal window.

Command:
clear

Description:
Exit from the terminal window.

Command:
exit
Description:
Display free and used system memory.

Command:
free

Description:
Display free and used system memory in bytes.

Command:
free -b

Description:
Display free and used system memory in kilobytes.

Command:
free -k

Description:
Display free and used system memory in megabytes.

Command:
Description:
Change user password.

Command:
`passwd`

Description:
Power-off the machine.

Command:
`shutdown`

Description:
Power-off the machine immediately.

Command:
`shutdown -h now`

Description:
Power-off the machine after 10 minutes.
Command:

```
shutdown -h +10
```

**Description:**

Print current working directory.

---

Command:

```
echo $PWD
```

**Description:**

Print previous working directory.

---

Command:

```
echo $OLDPWD
```

**Description:**

Executes the 11th command in command history.

---

Command:

```
!11
```
Description:
Reveals your command history.

Command:
```
history
```

Description:
Power off or reboot the Operating system.

Command:
```
sudo reboot
```

Description:
Display the IP address of the host.

Command:
```
ip address
```

Description:
List the size of files and directories.

Command:
```
ls -s
```
Description:

View mounted file systems.

Command:

```
mount
```

Description:

Display the information of disk usage of files and directories.

Command:

```
du
```

Description:

Tells you how long the system has been running.

Command:

```
uptime
```

Description:

Set current date as 02 Nov 1988.

Command:
date -- set 1998-11-02

Description:
Set current time as 12:11:02 IST.

Command:
date -- set 12:11:02

Description:
View and change the configuration of the network interfaces on the system.

Command:
ifconfig

Description:
Lists files.

Command:
ls

Description:
Report the process information.
Command:

```
ps
```

**Description:**

Display disk usage.

Command:

```
df
```

**Description:**

Display disk usage in gigabytes, megabytes, or kilobytes.

Command:

```
df -H
```

**Description:**

Delete every file and every directory.

Command:

```
rm -r *
```
**Description:**

Provides a quick overview of the currently running processes.

**Command:**

top

**Description:**

The system performs an immediate reboot.

**Command:**

reboot

**Description:**

Terminate processes without having to log out or reboot.

**Command:**

kill

**Description:**

Change the current working directory.

**Command:**

cd
**Description:**

Create a new session on the system.

**Command:**

`login`

**Description:**

List open files.

**Command:**

`lsof`

**Description:**

List USB devices.

**Command:**

`lsusb`

**Description:**

Check the status of the network services.

**Command:**
Description:

Start the network service.

Command:

```
service network start
```

Description:

Stop the network service.

Command:

```
service network stop
```

Description:

Restart the network service.

Command:

```
service network restart
```

Description:

Report information about the users currently on the machine and their processes.
Command: \texttt{w}

\textbf{Description:}

Display the current directory.

Command: \texttt{pwd}

\textbf{Description:}

Displays CPU architecture information (such as number of CPUs, threads, cores, sockets, and more).

Command: \texttt{lscpu}

\textbf{Description:}

Displays the number of processing units available to the current process.
**Description:**

The system performs an immediate reboot.

**Command:**

```
init 6
```

**Description:**

Power-off the machine.

**Command:**

```
init 0
```

**Description:**

List files by date.

**Command:**

```
ls -lrt
```

**Description:**

Report information about storage devices such as hard disks, flash drives etc.

**Command:**

```
lsblk
```
**Description:**

Show exit status of previous command.

**Command:**

```
echo $?
```

**Description:**

Lists a few useful info commands.

**Command:**

```
info
```

**Description:**

Prints current year's calendar.

**Command:**

```
cal -y
```

**Description:**

Check the status of all the services.

**Command:**

```
**service --status-all**

**Description:**
Display time in hh:mm:ss.

**Command:**
```
date +%T
```

**Description:**
Tells when the user last logged on and off and from where.

**Command:**
```
last -l username
```

**Description:**
Sort files and directories by extension name.

**Command:**
```
ls -X
```

**Description:**
Display the manual for the pwd command.
Command:

```
man pwd
```

**Description:**
Displays information about running processes in the form of a tree.

Command:

```
pstree
```

**Description:**
Resets your terminal.

Command:

```
reset
```

**Description:**
Displays What date is it this Friday.
**Description:**

Displays the size of each individual file.

---

**Command:**

```
du -a
```

---

**Description:**

Display information about the Advanced configuration and power Interface.

---

**Command:**

```
acpi
```

---

**Description:**

Takes you two folders back.

---

**Command:**

```
cd ../..
```

---

**Description:**

Takes you to the previous directory.

---

**Command:**

```
cd -
```

---
**Description:**

Displays a list of shell built-in commands.

---

**Command:**

`help`

---

**Description:**

Lists your last logins.

---

**Command:**

`last yourusername`

---

**Description:**

Create a new directory called myfiles.

---

**Command:**

`mkdir myfiles`

---

**Description:**

Remove the directory myfiles.

---

**Command:**
rmdir myfiles

**Description:**
Disable password for a specific user "root1".

**Command:**
```
passwd -d root1
```

**Description:**
Switch to user "root1".

**Command:**
```
sudo su root1
```

**Description:**
Exit from the terminal window.

**Command:**
```
logout
```

**Description:**
Creates a user "root1".
Command:

useradd "root1"

Description:

Assign password to user "root1".

Command:

passwd "root1"

Description:

Repeats the last command.

Command:

!!

Description:

Display Who you are logged in as.

Command:

whoami
**Description:**

Display the login name of the current user.

---

**Command:**

`logname`

---

**Description:**

Report the name of the kernel.

---

**Command:**

`uname`

---

**Description:**

Print the kernel version.

---

**Command:**

`uname -v`

---

**Description:**

Print the operating system.

---

**Command:**

`uname -o`
<table>
<thead>
<tr>
<th>Description</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report the machine hardware name.</td>
<td><code>uname -m</code></td>
</tr>
<tr>
<td>Print version information and exit.</td>
<td><code>uname --version</code></td>
</tr>
<tr>
<td>Print the kernel release.</td>
<td><code>uname -r</code></td>
</tr>
<tr>
<td>Report the network node hostname.</td>
<td></td>
</tr>
</tbody>
</table>
uname -n

**Description:**
Display all port connections (both TCP and UDP).

**Command:**
```
netstat -a
```

**Description:**
Display only TCP (Transmission Control Protocol) port connections.

**Command:**
```
netstat -at
```

**Description:**
Display only UDP (User Datagram Protocol) port connections.

**Command:**
```
netstat -au
```

**Description:**
Display all active listening ports.
Command:

```
netstat -I
```

**Description:**

Display all active listening TCP ports.

Command:

```
netstat -It
```

**Description:**

Display all active listening UDP ports.

Command:

```
netstat -lu
```

**Description:**

Reveal all the information about the current user (user id, username, group id, group name etc.).
**Description:**

Reveal all the information about the user "root1" (user id, username, group id, group name etc.).

**Command:**

```
id root1
```

**Description:**

Print the machine's architecture.

**Command:**

```
arch
```

**Description:**

Display the list of available fonts.

**Command:**

```
fc-list
```

**Description:**

Create two directories (myfiles, files).

**Command:**

```
mkdir myfiles files
```
Description:
install apache (CentOS).

Command:
yum install httpd

Description:
install apache (Ubuntu).

Command:
apt install httpd

Description:
upgrade apache (CentOS).

Command:
yum update httpd

Description:
upgrade apache (Ubuntu).

Command:
Description:

uninstall apache (CentOS).

Command:
yum remove httpd

Description:

uninstall apache (Ubuntu).

Command:
apt remove httpd

Description:

Display usage summary for the command (date).

Command:
date --help

Description:

List active connections to/from system.
Command:

ss -tup

Description:
List internet services on a system.

Command:

ss -tupl

Description:
Display all active UNIX listening ports.

Command:

netstat -lx

Description:
Display all the active interfaces details.

Command:

ifconfig
Description:

Display information of all network interfaces.

Command:

```
ifconfig -a
```

Description:

Compare the contents of two files (1.txt, 2.txt).

Command:

```
diff 1.txt 2.txt
```

Description:

Tells you how many lines, words, and characters there are in a file (1.txt).

Command:

```
w 1.txt
```

Description:

Compresses file (1.txt), so that it take up much less space.

Command:

```
gzip 1.txt
```
**Description:**

Uncompresses file (1.txt) compressed by gzip.

**Command:**

`gunzip 1.txt`

**Description:**

Examine the contents of the file (1.txt).

**Command:**

`cat 1.txt`

**Description:**

Display calendar.

**Command:**

`ncal`

**Description:**

Removes the file (1.txt).

**Command:**
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>rm 1.txt</code></td>
<td>Rename a file named 1.txt to 0.txt.</td>
</tr>
<tr>
<td><code>mv 1.txt 0.txt</code></td>
<td>Replace the contents of 0.txt with that of 1.txt.</td>
</tr>
<tr>
<td><code>touch test.txt</code></td>
<td>Create an empty file (test.txt).</td>
</tr>
<tr>
<td><code>cp 1.txt 0.txt</code></td>
<td>Print the last 10 lines of a file (1.txt).</td>
</tr>
</tbody>
</table>
Command:

`tail 1.txt`

**Description:**

Print N number of lines from the file (1.txt).

Command:

`tail -n N 1.txt`

**Description:**

Prints the number of words in a file (1.txt).

Command:

`wc -w 1.txt`

**Description:**

Prints the number of characters from a file (1.txt).

Command:

`wc -m 1.txt`
**Description:**

Prints the length of the longest line in a file (1.txt).

**Command:**

```bash
wc -L 1.txt
```

**Description:**

Print information about usb ports, graphics cards, network adapters etc.

**Command:**

```bash
lspci
```

**Description:**

View contents of a file (1.txt).

**Command:**

```bash
less 1.txt
```

**Description:**

Display calendar (last month, current month, and next month).

**Command:**

```bash
cal -3
```
**Description:**

Compare the contents of three files (1.txt, 2.txt, 3.txt) line by line.

---

Command:

```
diff3 1.txt 2.txt 3.txt
```

---

**Description:**

Compare two files (1.txt, 2.txt) line-by-line.

---

Command:

```
comm 1.txt 2.txt
```

---

**Description:**

Perform byte-by-byte comparison of two files (1.txt, 2.txt).

---

Command:

```
cmp 1.txt 2.txt
```

---

**Description:**

Prints the CRC checksum and byte count for the file "myfiles.txt".

---

Command:
Description:

Append contents of files (1.txt, 2.txt) into one file (0.txt).

Command:

```
cat 1.txt 2.txt > 0.txt
```

Description:

Append contents of files (1.txt, 2.txt, 3.txt) into one file (0.txt).

Command:

```
sed  r 1.txt 2.txt 3.txt > 0.txt
```

Description:

Append contents of files (1.txt, 2.txt, 3.txt) into one file (0.txt).

Command:

```
sed  h 1.txt 2.txt 3.txt > 0.txt
```

Description:

Append contents of files (1.txt, 2.txt, 3.txt) into one file (0.txt).
Command:

```
!sed -n  p 1.txt 2.txt 3.txt > 0.txt
```

Shortcuts:

| ctrl+c | Halts the current command |
| ctrl+z | Stops the current command |
| ctrl+d | Logout the current session |
| ctrl+w | Erases one word in the current line |
| ctrl+u | Erases the whole line |
| ctrl+r | Type to bring up a recent command |

Description:

Writes contents of a file (0.txt) to output, and prepends each line with line number.

Command:

```
!nl 0.txt
```

Description:

Create a empty file (test1.txt) inside a directory (test).

Command:

```
!mkdir test
cd test
pwd
touch test1.txt
```
**Description:**

Gather information about hardware components such as CPU, disks, memory, USB controllers etc.

**Command:**

```bash
sudo lshw
```

**Description:**

Gather information about file system partitions.

**Command:**

```bash
sudo fdisk -l
```

**Description:**

Displays the line (good morning) in which the string (good) is found in the file (1.txt).

**Command:**

```bash
grep good 1.txt
```

**Description:**

Append contents of files (1.txt, 2.txt, 3.txt) into one file (0.txt) using for loop.
Command:

for i in {1..3}; do cat "$i.txt" >> 0.txt; done

Description:

Search for files (test.txt, test1.txt, test2.txt, test.php, test.html) in a directory as well as its subdirectories.

Command:

find test*

Description:

Displays status related to a file (1.txt).

Command:

stat 1.txt

###

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vi</td>
<td>Open vi editor</td>
</tr>
<tr>
<td>i</td>
<td>Go to Insert mode</td>
</tr>
<tr>
<td>a =20; b =64;</td>
<td></td>
</tr>
<tr>
<td>print (a + b);</td>
<td></td>
</tr>
<tr>
<td>Hit Escape to return to Normal mode.</td>
<td></td>
</tr>
<tr>
<td>:w hello.py</td>
<td>Save text</td>
</tr>
<tr>
<td>:q</td>
<td>Quit</td>
</tr>
<tr>
<td>python hello.py</td>
<td>Print the output:84</td>
</tr>
</tbody>
</table>

Description:
Download the file (file.txt) from url "http://website.com/files/file.txt".

Command:

```
wget http://website.com/files/file.txt
```

**Description:**

Display host's numeric ID in hexadecimal format.

Command:

```
hostid
```

**Description:**

Display file type of the file (myfiles.txt).

Command:

```
file myfiles.txt
```

**Description:**

Create a file (myfiles.txt) containing a text (Hello World).

Command:

```
echo 'Hello World' > myfiles.txt
```
Description:

Create a file (myfiles.txt) containing a text (Hello World).

Command:

```bash
printf 'Hello World' > myfiles.txt
```

Description:

Display IP address of the hostname.

Command:

```bash
hostname -i
```

Description:

Add a new line of text to an existing file (1.txt).

Command:

```bash
echo "Hello world!" >> 1.txt
echo "this is 2nd line text" >> 1.txt
echo "last line!" >> 1.txt
```

Description:

Displays a single line description about a command (cal).
Command:

```
whatis cal
```

###
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vi</td>
<td>Open vi editor</td>
</tr>
<tr>
<td>i</td>
<td>Go to Insert mode</td>
</tr>
<tr>
<td>Type some text.</td>
<td></td>
</tr>
<tr>
<td>Hit Escape to return to Normal mode.</td>
<td></td>
</tr>
<tr>
<td>:w test.txt</td>
<td>Save text</td>
</tr>
<tr>
<td>:q</td>
<td>Quit</td>
</tr>
<tr>
<td>:q!</td>
<td>Quit without saving</td>
</tr>
</tbody>
</table>

###
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vi</td>
<td>Open vi editor</td>
</tr>
<tr>
<td>i</td>
<td>Go to Insert mode</td>
</tr>
<tr>
<td>$name = &quot;Paul&quot;;</td>
<td></td>
</tr>
<tr>
<td>print &quot;$name&quot;;</td>
<td></td>
</tr>
<tr>
<td>Hit Escape to return to Normal mode.</td>
<td></td>
</tr>
<tr>
<td>:w hello.pl</td>
<td>Save text</td>
</tr>
<tr>
<td>:q</td>
<td>Quit</td>
</tr>
<tr>
<td>perl hello.pl</td>
<td>Print the output: Paul</td>
</tr>
</tbody>
</table>

###
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vi</td>
<td>Open vi editor</td>
</tr>
<tr>
<td>i</td>
<td>Go to Insert mode</td>
</tr>
<tr>
<td>echo &quot;What is your name?&quot;</td>
<td></td>
</tr>
<tr>
<td>read PERSON</td>
<td></td>
</tr>
<tr>
<td>echo &quot;Hello, $PERSON&quot;</td>
<td></td>
</tr>
<tr>
<td>Hit Escape to return to Normal mode.</td>
<td></td>
</tr>
<tr>
<td>:w hello.sh</td>
<td>Save text</td>
</tr>
<tr>
<td>:q</td>
<td>Quit</td>
</tr>
<tr>
<td>sh hello.sh</td>
<td>Output:</td>
</tr>
<tr>
<td></td>
<td>What is your name?</td>
</tr>
<tr>
<td></td>
<td>If you enter: Zara Ali</td>
</tr>
<tr>
<td></td>
<td>Hello, Zara Ali</td>
</tr>
</tbody>
</table>

Description:

Check the network connectivity between host (your connection) and server (Google server).
Command:

\texttt{ping google.com}

\textbf{Description:}

Find the location of source/binary file of a command (cal).

Command:

\texttt{whereis cal}

\textit{41 C Exercises for Beginners}

\section*{Question 1}

\textbf{Question:}

Write a program to print Hello World!

\textbf{Solution:}

\begin{verbatim}
#include<stdio.h>
int main()
{
    printf("Hello, world!");
    return 0;
}
\end{verbatim}

\section*{Question 2}
Question:

Write a program to find the area of a circle.

Solution:

```c
#include<stdio.h>
int main()
{
    int r, area;
    r = 2;
    area = 4 * 3.14 * r * r;
    printf("The area of the circle = %d", area);
    return 0;
}
```

Question 3

Question:

Write a program to find the sum of two numbers.

Solution:

```c
#include<stdio.h>
int main()
{
    int a, b, sum;
    a=1;
    b=2;
    sum = a + b;
    printf("The sum of a and b = %d", sum);
    return 0;
}
```
Question:

Write a program to find the square of a number.

Solution:

```c
#include<stdio.h>
#include<math.h>
int main()
{
    int a, b;
    a = 2;
    b = pow((a), 2);
    printf("The square of a = %d", b);
    return 0;
}
```

Question 5

Question:

Write a program to find the greatest of two numbers.

Solution:

```c
#include<stdio.h>
int main()
{
    int a, b;
    a = 2;
    b = 3;
    if(a>b)
    {
        printf("a is greater than b");
    }
    else
    {
        printf("b is greater than a");
    }
    return 0;
}
```
Question 6

Question:

Write a program to print the average of the elements in the array.

Solution:

```c
#include<stdio.h>
int main()
{
    int i, avg, sum = 0;
    for(i=0; i<5; i++)
        sum = sum + num[i];
    avg = sum/5;
    printf("Sum of the Elements in the array = %d", sum);
    printf("Average of the elements in the array= %d", avg);
    return 0;
}
```

Question 7

Question:

Write a program such that a Switch (case) allows to make a decision from the number of choices, i.e., from the number of cases.

Solution:

```c
#include<stdio.h>
int main()
{
    char ch;
    printf("Enter any character:");
    scanf("%c", &ch);
    switch(ch)
    {
```
```c
#include<stdio.h>

int main()
{
    int x, y, *p, *q;
    printf("Enter any integer:");
    scanf("%d", &x);
    printf("Enter any integer:");
    scanf("%d", &y);
    p = &x;
    q = &y;
    if(*p>*q)
    {
      printf("x is greater than y");
    }
    if(*q>*p)
    {
      printf("y is greater than x");
    }
    return 0;
}
```

Question 8

Question:

Write a program to find the greatest of two numbers using pointers.
Question 9

Question:

Write a program to print the address of x and the value assigned to x.

Solution:

```c
#include <stdio.h>
int main()
{
    int x, *p;
    x = 1;
    p = &x;
    printf("The address of the variable x =\d", p);
    printf("The value of the variable x =\d", *p);
    return 0;
}
```

Question 10

Question:

Write a program to print the first 10 numbers starting from one together with their squares and cubes.

Solution:

```c
#include<stdio.h>
int main()
{
    int i;
    for ( i=1; i<=10; i++)
        printf("Number=%d its square=%d its cube=%d\n", i , i*i, i*i*i);
    return 0;
}
```
Question 11

Question:

Write a program:

If you enter a character M
Output must be: ch = M.

Solution:

```c
#include<stdio.h>

int main()
{
    char M;
    printf("Enter any character: ");
    scanf("%c", &M);
    printf("ch=%c", M);
    return 0;
}
```

Question 12

Question:

Write a program to print the multiplication table of a number.

Solution:

```c
#include<stdio.h>

int main()
{
    int n, i;
    printf("Enter any number: ");
    scanf("%d", &n);
    ```
Question 13

Question:

Write a program to print the product of the first 10 digits.

Solution:

```c
#include<stdio.h>
int main()
{
    int i, product = 1;
    for( i=1; i<=10; i++)
        product *= i;
    printf("The product of the first 10 digits is %d\n", product);
    return 0;
}
```

Question 14

Question:

Write a program to print whether the given number is positive or negative.

Solution:

```c
#include<stdio.h>
int main()
{
    int a;
    a = -35;
    if(a>0)
```
Question 15

Question:

Write a program to check the equivalence of two numbers.

Solution:

```c
#include<stdio.h>
int main()
{
    int x, y;
    printf("Enter any number:");
    scanf("%d", &x);
    printf("Enter any number:");
    scanf("%d", &y);
    if(x-y==0)
    {
        printf("The two numbers are equivalent");
    }
    else
    {
        printf("The two numbers are not equivalent");
    }
    return 0;
}
```

Question 16

Question:
Write a program to print the remainder of two numbers.

Solution:

```c
#include<stdio.h>
int main()
{
    int a, b, c;
    printf("Enter any number:");
    scanf("%d", &a);
    printf("Enter any number:");
    scanf("%d", &b);
    c = a%b;
    printf("The remainder of a and b = %d", c);
    return 0;
}
```

Question 17

Question:

Write a program to print the given number is even or odd.

Solution:

```c
#include<stdio.h>
int main()
{
    int a;
    printf("Enter any number:");
    scanf("%d", &a);
    if(a%2 == 0)
    {
        printf("The number is even");
    }
    else
    {
        printf("The number is odd");
    }
    return 0;
}
```
Question 18

Question:
Write a program to print the characters from A to Z.

Solution:

```c
#include<stdio.h>
int main()
{
    char a;
    for( a='A'; a<='Z'; a++)
        printf("%c\n", a);
    return 0;
}
```

Question 19

Question:
Write a program to find the incremented and decremented values of two numbers.

Solution:

```c
#include<stdio.h>
int main()
{
    int a, b, c, d, e, f;
    a = 10;
    b=12;
    c=a+1;
    d=b+1;
    e=a-1;
    f=b-1;
    printf("The incremented value of a =%d", c);
    printf("The incremented value of b =%d", d);
    printf("The decremented value of a =%d", e);
    printf("The decremented value of b =%d", f);
}
```
Question 20

Question:
Write a program to calculate the simple interest.

Solution:

```c
#include<stdio.h>
int main()
{
    int P, T, R, SI;
    P = 1000;
    T = 2;
    R = 3;
    SI = P*T*R/100;
    printf("The simple interest = \$d", SI);
    return 0;
}
```

Question 21

Question:
Write a program to Find the largest of three numbers.

Solution:

```c
#include<stdio.h>
int main()
{
    int a, b, c;
    printf("Enter any number: ");
```
```c
scanf("%d", &a);
printf("Enter any number: ");
scanf("%d", &b);
printf("Enter any number: ");
scanf("%d", &c);
if (a > b && a > c)
{
    printf("%d is greater than %d and %d", a, b, c);
}
else if (b > a && b > c)
{
    printf("%d is greater than %d and %d", b, a, c);
}
else
{
    printf("%d is greater than %d and %d", c, b, a);
}
return 0;
}
```

---

**Question 22**

**Question:**

Write a program to print the factorial of the entered number.

---

**Solution:**

```c
#include<stdio.h>
int main()
{
    int i, n, fact = 1;
    printf("Enter any number:");
    scanf("%d", &n);
    for(i=1; i<=n; i++)
        fact = fact * i;
    printf("Entered number is: %d", n);
    printf("The factorial of the entered number %d is: %d", n, fact);
    return 0;
}
```
Question 23

Question:

Write a program to print the length of the entered string.

Solution:

```c
#include<stdio.h>
#include<string.h>
int main()
{
    char ch[4];
    printf("Enter any word: ");
    scanf("%c", &ch);
    printf("The length of the string = %d", strlen(ch));
    return 0;
}
```

Question 24

Question:

Write a program to print the ASCII value of the entered character.

Solution:

```c
#include<stdio.h>
int main()
{
    char ch = 'A';
    printf("The ASCII value of ch is: %d", ch);
    return 0;
}
```
Question 25

Question:

Write a program to check whether the entered character is a lower case letter or not.

Solution:

```c
#include<stdio.h>
int main()
{
    char ch = 'a';
    if(islower(ch))
        printf("you have entered the lower case letter");
    else
        printf("you have entered the upper case letter");
    return 0;
}
```

Question 26

Question:

Write a program to check whether the entered character is a upper case letter or not.

Solution:

```c
#include<stdio.h>
int main()
{
    char ch = 'a';
    if(isupper(ch))
        printf("you have entered the upper case letter");
    else
        printf("you have entered the lower case letter");
    return 0;
}
```
Question 27

Question:
Write a program to convert the lower case letter to upper case letter.

Solution:

```c
#include<stdio.h>
int main()
{
    char ch = 'a';
    char b = toupper(ch);
    printf("lower case letter %c is converted to upper case letter %c", ch, b);
    return 0;
}

```

Question 28

Question:
Write a program to print the output:

Einstein [0] = E
Einstein [1] = I
Einstein [2] = N
Einstein [3] = S
Einstein [5] = E
Einstein [6] = I
Einstein [7] = N
Solution:

```c
#include<stdio.h>
int main()
{
    int i;
    char name [8] = {'E', 'I', 'N', 'S', 'T', 'E', 'I', 'N'};
    for(i=0; i<8; i++)
        printf("\n Element[%d] = %c", i, name[i]);
    return 0;
}
```

Question 29

Question:

Write a program to print the output:

- Name of the book = B
- Price of the book = 135.00
- Number of pages = 300
- Edition = 8

using structures.

Solution:

```c
#include<stdio.h>
int main()
{
    struct book {
        char name;
        float price;
        int pages;
        int edition;
    };
    struct book bl;
    bl.name = 'B';
    bl.price = 135.00;
    bl.pages = 300;
    bl.edition = 8;
    printf("\n Name of the book = %c", bl.name);
    printf("\n Price of the book = %f", bl.price);
    printf("\n Number of pages = %d", bl.pages);
```
Question 30

Question:

Write a program to find square of a number using functions.

Solution:

```c
#include<stdio.h>
int square();
int main()
{
    int answer;
    answer = square();
    printf("Square of the given number=%d", answer);
    return 0;
}

int square()
{
    int x;
    printf("Enter any integer:");
    scanf("%d", &x);
    return x*x;
}
```

Question 31

Question:

Write a program To print "hello world" 10 times.

Solution:
```c
#include<stdio.h>
int main()
{
    int i;
    for (i =1; i<=10; i++)
        printf("hello world \n");
    return 0;
}
```

**Question 32**

**Question:**

Write a program to print first 5 numbers using do while loop statement.

**Solution:**

```c
#include<stdio.h>
int main()
{
    int i =1;
    do
    {
        printf("%d\n", i++);
    } while (i<=5);
    return 0;
}
```

**Question 33**

**Question:**

Write a program to print the output:

```c
body [b] = b
body [o] = o
body [d] = d
```
body [y] = y

Solution:

```c
#include <stdio.h>
int main()
{
    char i;
    char body[4] = {'b', 'o', 'd', 'y'};
    for (i = 0; i < 4; i++)
        printf (“\n body[%c] = %c”, body[i], body[i]);
    return 0;
}
```

Question 34

Question:

What will be the output of the below program:

```c
#include<stdio.h>
#include<stdlib.h>
int main()
{
    printf(“linux\n”);
    exit (0);
    printf(“php\n”);
    return 0;
}
```

Solution:

```
linux
```

Question 35

Question:
Write a program to check whether a character is an alphabet or not.

Solution:

```c
#include <stdio.h>
#include <ctype.h>
int main() {
    int a = 'a';
    if (isalpha(a)) {
        printf("The character a is an alphabet");
    } else {
        printf("The character a is not an alphabet");
    }
    return 0;
}
```

Question 36

Question:

Write a program to calculate the discounted price and the total price after discount

Given:

- If purchase value is greater than 1000, 10% discount
- If purchase value is greater than 5000, 20% discount
- If purchase value is greater than 10000, 30% discount.

Solution:

```c
#include <stdio.h>
int main() {
    double PV;
    printf("Enter purchased value:");
    scanf("%lf", &PV);
    if (PV>1000) {
        printf("\nDiscount=\%lf", PV* 0.1);
    }
```
Question 37

Question:

Write a program to print the first ten natural numbers using while loop statement.

Solution:

```c
#include<stdio.h>
int main()
{
    int i = 1;
    while (i<=10)
    {
        printf("%d\n", i++);
    }
    return 0;
}
```

Question 38

Question:

What will be the output of the below program:
```c
#include <stdio.h>
int main()
{
    int i;
    for (i=1; i<=5; i++)
    {
        if (i==3)
            continue;
    }
    printf("%d
", i);
    return 0;
}
```

Solution:

```
1
2
4
5
```

Question 39

Question:

Write a program to find the size of an array.

Solution:

```c
#include <stdio.h>
int main()
{
    int num [] = {11, 22, 33, 44, 55, 66};
    int n;

    /* Calculating the size of the array with this formula.
    * n = sizeof(array_name) / sizeof(array_name[0])
    * This is a universal formula to find number of elements in
    * an array, which means it will work for arrays of all data
    * types such as int, char, float etc.
    */
    n = sizeof(num) / sizeof(num [0]);
    printf("Size of the array is: %d\n", n);
    return 0;
```
Question 40

Question:

What would be the output of the following programs:

```c
#include <stdio.h>
int main()
{
    int i;
    for (i=1; i<=5; i++)
    {
        if (i==3)
        {
            break;
        }
        printf("%d\n", i);
    }
    return 0;
}
```

Solution:

1
2

```c
#include <stdio.h>
int main()
{
    int i;
    for (i=1; i<=5; i++)
    {
        if (i==3)
        {
            goto HAI;
        }
        printf("\n%d ", i);
    }  
    HAI : printf("\n Linux");
}
```

Solution:
```c
#include<stdio.h>
int main()
{
    int i = 54;
    int y = i<<1;
    printf("The value of y = %d", y);
    return 0;
}

Solution:

The value of y = 108
```

```c
#include<stdio.h>
int main()
{
    int i = 54;
    int y = i>>1;
    printf("The value of y = %d", y);
    return 0;
}

Solution:

The value of y = 27
```

```c
#include<stdio.h>
#include<stdlib.h>
int main()
{
    int a, b;
    a= - 2;
    b= abs(a);
    printf("Absolute value = %d", b);
    return 0;
}

Solution:

Absolute value = 2
```

```c
#include <stdio.h>
int main()
{
    for ( ; ; )
```
Solution:

This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever. .......

#include<stdio.h>
int main()
{
    printf("Hello,world!");
    return 0;
    printf("Hello,world!");
}

Solution:

Hello,world!

Question 41

Question:
Write a program to check whether the person is a senior citizen or not.

Solution:

#include<stdio.h>
int main()
{
    int age;
    printf("Enter age:");
    scanf("%d", &age);
    if(age>=60)
42 C++ Exercises for Beginners

Question 1

Question:
Write a program to print Hello World!

Solution:

```cpp
#include<iostream>
int main()
{
    std::cout<<"Hello World!";
    return 0;
}
```

```
#include<iostream>
using namespace std;
int main()
{
    cout<<"Hello World!";
    return 0;
}
```

---

Question 2

Question:
Write a program to find the area of a circle.

Solution:

```cpp
#include<iostream>
using namespace std;
int main()
{
 float r, area;
cout<<"Enter any number:";
cin>>r;
area = 4 * 3.14 * r * r;
cout<<"The area of the circle = " << area;
return 0;
}
```

Question 3

Question:

Write a program to find the sum of two numbers.

Solution:

```cpp
#include<iostream>
using namespace std;
int main()
{
 float a, b, sum;
cout<<"Enter any two numbers:";
cin>>a;
cin>>b;
sum = a + b;
cout<<"The sum of a and b = " << sum;
return 0;
}
```
Question:

Write a program to find the square of a number.

Solution:

```cpp
#include<iostream>
using namespace std;
int main()
{
    int a, b;
a = 2;
b = a * a;
cout<<"The square of a = " << b;
return 0;
}
```

Question 5

Question:

Write a program to find the greatest of two numbers.

Solution:

```cpp
#include<iostream>
using namespace std;
int main()
{
    int a, b;
a = 2;
b = 3;
if (a > b)
{
    cout << "a is greater than b";
}
else
{
    cout << "b is greater than a";
}
return 0;
}
```
Question 6

Question:
Write a program to print the average of the elements in the array.

Solution:

```cpp
#include<iostream>
using namespace std;
int main()
{
    int i, avg, sum = 0;
    for(i=0; i<5; i++)
        sum = sum + num [i];
    avg = sum/5;
    cout << "Sum of the Elements in the array = " << sum << endl;
    cout << "Average of the elements in the array= " << avg << endl;
    return 0;
}
```

Question 7

Question:
Write a program such that a Switch (case) allows to make a decision from the number of choices, i.e., from the number of cases.

Solution:

```cpp
#include<iostream>
using namespace std;
int main()
{
    char ch;
    cout << "Enter any character:"; 
    cin >> ch;
    return 0;
}
```
switch(ch) {
  case 'R':
    cout<<"Red";
    break;
  case 'W':
    cout<<"White";
    break;
  case 'Y':
    cout<<"Yellow";
    break;
  case 'G':
    cout<<"Green";
    break;
  default:
    cout<<"Error";
    break;
}
return 0;

Question 8

Question:

Write a program to find the greatest of two numbers using pointers.

Solution:

#include<iostream>
using namespace std;
int main() {
    int x, y, *p, *q;
    cout<<"Enter any integer:";
    cin>>x;
    cout<<"Enter any integer:"
    cin>>y;
    p = &x;
    q = &y;
    if (*p>*q) {
        cout<<"x is greater than y";
    } else {
        cout<<"y is greater than x";
    }
Question 9

Question:
Write a program to print the address of x and the value assigned to x.

Solution:

```cpp
#include<iostream>
using namespace std;
int main()
{
    int x, *p;
    cout<<"Enter any integer:";
    cin>>x;
p = &x;
    cout<<"The address of the variable x = "<< p << endl;
    cout<<"The value of the variable x = "<< *p << endl;
    return 0;
}
```

Question 10

Question:
Write a program to print the first 10 numbers starting from one together with their squares and cubes.

Solution:

```cpp
#include<iostream>
using namespace std;
int main()
{
    // Code here
}
```
```cpp
{ 
    int i;
    for( i=1; i<=10; i++)
    cout<<"number = " << i << " its square = " << i*i << " its cube = " << i*i*i<< endl;
    return 0;
}
```

---

### Question 11

**Question:**

Write a program:

If you enter a character M

Output must be: ch = M.

---

**Solution:**

```cpp
#include<iostream>
using namespace std;
int main()
{
    char M;
    cout<<"Enter any character:"
    cin>>M;
    cout<<"ch= " << M;
    return 0;
}
```

---

### Question 12

**Question:**

Write a program to print the multiplication table of a number.

---

**Solution:**
Question 13

Question:

Write a program to print the product of the first 10 digits.

Solution:

```cpp
#include<iostream>
using namespace std;
int main()
{
    int i, product = 1;
    for( i=1; i<=10; i++)
        product = product * i;
    cout<<"The product of the first 10 digits = " << product;
    return 0;
}
```

Question 14

Question:

Write a program to print whether the given number is positive or negative.

Solution:

```cpp
#include<iostream>
using namespace std;
int main()
{
    int n, i;
    cout<<"Enter any number:\n";
    cin>>n;
    for( i=1; i<=5; i++)
        cout<< n <<" * "<< i <<" = " << n*i <<endl;
    return 0;
}
```
```cpp
#include<iostream>
using namespace std;
int main()
{
    int a;
a = -35;
    if(a>0)
    {
        cout<<"Number is positive";
    } else 
    {
        cout<<"Number entered is negative";
    }
    return 0;
}
```

**Question 15**

**Question:**

Write a program to check the equivalence of two numbers.

**Solution:**

```cpp
#include<iostream>
using namespace std;
int main()
{
    int x, y;
cout<<"Enter any number:";
cin>>x;
cout<<"Enter any number:";
cin>>y;
if(x-y==0)
    {
        cout<<"The two numbers are equivalent";
    } else 
    {
        cout<<"The two numbers are not equivalent";
    }
    return 0;
}
```
Question 16

Question:
Write a program to print the remainder of two numbers.

Solution:

```cpp
#include<iostream>
using namespace std;
int main()
{
    int a, b, c;
    cout<<"Enter any number:";
    cin>>a;
    cout<<"Enter any number:";
    cin>>b;
    c = a % b;
    cout<<"The remainder of a and b = "<< c;
    return 0;
}
```

Question 17

Question:
Write a program to print the given number is even or odd.

Solution:

```cpp
#include<iostream>
using namespace std;
int main()
{
    int a;
    cout<<"Enter any number:";
    cin>>a;
    if(a%2 == 0)
    {
```
```cpp
#include<iostream>
using namespace std;
int main()
{
    char a = 'A';
    while (a<='Z')
    {
        cout<<"\n"<< a++;
    }
    return 0;
}
```

---

**Question 18**

**Question:**
Write a program to print the characters from A to Z.

**Solution:**

```cpp
#include<iostream>
using namespace std;
int main()
{
    char a = 'A';
    while (a<='Z')
    {
        cout<<"\n"<< a++;
    }
    return 0;
}
```

---

**Question 19**

**Question:**
Write a program to find the incremented and decremented values of two numbers.

**Solution:**

```cpp
#include<iostream>
```
using namespace std;
int main()
{
    int a, b, c, d, e, f;
a = 10;
b=12;
c=a+1;
d=b+1;
e=a-1;
f=b-1;
cout<<"The incremented value of a = "<< c << endl;
cout<<"The incremented value of b = "<< d << endl;
cout<<"The decremented value of a = "<< e << endl;
cout<<"The decremented value of b = "<< f << endl;
return 0;
}

Question 20

Question:
Write a program to calculate the simple interest.

Solution:

#include<iostream>
using namespace std;
int main()
{
    int P, T, R, SI;
cout<<"Enter principal amount:";
cin>>P;
cout<<"Enter time:";
cin>>T;
cout<<"Enter rate of interest:";
cin>>R;
SI = P*T*R/100;
cout<<"The simple interest = "<<SI;
return 0;
}
Question 21

Question:

Write a program to Find the largest of three numbers.

Solution:

```cpp
#include<iostream>
using namespace std;
int main()
{
  int a, b, c;
  cout<<"Enter any number:"
  cin>>a;
  cout<<"Enter any number:"
  cin>>b;
  cout<<"Enter any number:"
  cin>>c;
  if(a>b&&a>c)
  {cout<< a<<" is greater than "<< b<<" and "<<c;
  }else if (b>a&&b>c)
  {cout<< b<<" is greater than "<< a <<" and "<<c;
  }else
  {
  cout<< c<<" is greater than "<< b<<" and "<< a;
  }
  return 0;
}
```

Question 22

Question:

Write a program to print the factorial of the entered number.
Question 23

Question:

Write a program to print the length of the entered string.

Solution:

```c++
#include<iostream>
#include<string.h>
using namespace std;
int main()
{
    char ch[4];
    cout<<"Enter any word: ";
    cin>>ch;
    cout<<"The length of the string = "<< strlen(ch); 
    return 0;
}
```

Question 24

Question:
Write a program to print the ASCII value of the entered character.

Solution:

```cpp
#include <iostream>
using namespace std;
int main()
{
    char c;
    cout << "Enter a character: ";
    cin >> c;
    cout << "ASCII Value of " << c << " is " << int(c);
    return 0;
}
```

---

**Question 25**

**Question:**

Write a program to check whether the entered character is a lower case letter or not.

**Solution:**

```cpp
#include <iostream>
using namespace std;
int main()
{
    char ch = 'a';
    if(islower(ch))
        cout << "you have entered the lower case letter";
    else
        cout << "you have entered the upper case letter";
    return 0;
}
```

---

**Question 26**

**Question:**
Write a program to check whether the entered character is a upper case letter or not.

---

**Solution:**

```cpp
#include<iostream>
using namespace std;
int main()
{
    char ch = 'a';
    if(isupper(ch))
        cout<<"you have entered the upper case letter";
    else
        cout<<"you have entered the lower case letter";
    return 0;
}
```

---

**Question 27**

**Question:**

Write a program to convert the lower case letter to upper case letter.

---

**Solution:**

```cpp
#include<iostream>
using namespace std;
int main()
{
    char ch = 'a';
    char b = toupper(ch);
    cout<<" lower case letter "<<ch<<" is converted to upper case letter "<<b;
    return 0;
}
```

---

**Question 28**

**Question:**
Write a program to print the output:

Einstein [0] = E
Einstein [1] = I
Einstein [2] = N
Einstein [3] = S
Einstein [5] = E
Einstein [6] = I
Einstein [7] = N

---

Solution:

```cpp
#include<iostream>
using namespace std;

int main()
{
    int i;
    char name [8] = {'E', 'I', 'N', 'S', 'T', 'E', 'I', 'N'};
    for(i=0; i<8; i++)
        cout<<"Element ["<< i <<"] = "<< name[i] << endl;
    return 0;
}
```

---

**Question 29**

**Question:**

Write a program to print the output:

Name of the book = B
Price of the book = 135.00
Number of pages = 300
Edition = 8
using structures.
Solution:

```cpp
#include<iostream>
using namespace std;
int main()
{
    struct book {
        char name;
        float price;
        int pages;
        int edition;
    };
    struct book bl = {'B', 135.00, 300, 8};
    cout<<"Name of the book = "<< bl.name<< endl;
    cout<<"Price of the book = "<< bl.price<<endl;
    cout<<"Number of pages = "<< bl.pages<<endl;
    return 0;
}
```

Question 30

Question:

Write a program to find square of a number using functions.

Solution:

```cpp
#include<iostream>
using namespace std;
int square();
int main()
{
    int answer;
    answer = square();
    cout<<"Square of the given number = "<< answer;
    return 0;
}
int square()
{
    int x;
    cout<<"Enter any integer:"
    cin>>x;
    return x*x;
}```
Question 31

Question:
Write a program To print "hello world" 10 times.

Solution:

```cpp
#include<iostream>
using namespace std;

int main()
{
    int i;
    for (i = 1; i<=10; i++)
        cout<<"\n hello world";
    return 0;
}
```

Question 32

Question:
Write a program to print first 5 numbers using do while loop statement.

Solution:

```cpp
#include<iostream>
using namespace std;

int main()
{
    int i=1;
    do
    {
        cout<<" \n i= " << i++;
    } while (i<=5);
    return 0;
}
```
Question 33

Question:

Write a program to print the output:

- `body[b] = b`
- `body[o] = o`
- `body[d] = d`
- `body[y] = y`

Solution:

```cpp
#include <iostream>
using namespace std;

int main()
{
    char i;
    char body[4] = {'b', 'o', 'd', 'y'};
    for (i=0; i<4; i++)
        cout << "\n body [" << body[i] << "] = " << body[i] << endl;
    return 0;
}
```

Question 34

Question:

What will be the output of the below program:

```cpp
#include <iostream>
using namespace std;

int main()
{
```
}
cout<<"linux\n"; exit(0); cout<<"php\n"; return 0; }

Solution:

linux

---

**Question 35**

**Question:**

Write a program to check whether a character is an alphabet or not.

**Solution:**

```cpp
#include <iostream>
using namespace std;
int main()
{
    int a = 97;
    if(isalpha(a))
    {
        cout<<"The character a is an alphabet";
    }
    else
    {
        cout<<"The character a is not an alphabet";
    }
    return 0;
}
```

---

**Question 36**

**Question:**
Write a program to calculate the discounted price and the total price after discount

Given:

If purchase value is greater than 1000, 10% discount
If purchase value is greater than 5000, 20% discount
If purchase value is greater than 10000, 30% discount.

Solution:

```c++
#include<iostream>
using namespace std;
int main()
{
 double PV;
 cout<<"Enter purchased value:";
 cin>>PV;
 if (PV>1000)
 {
  cout<<"Discount = "<< PV* 0.1 << endl;
  cout<<"Total= "<< PV - PV* 0.1 << endl;
 } 
 else if (PV>5000)
 { 
  cout<<"Discount = "<< PV* 0.2 << endl;
  cout<<"Total= "<< PV - PV* 0.1 << endl;
 } 
 else 
 { 
  cout<<"Discount = "<< PV* 0.3 << endl;
  cout<<"Total= "<< PV - PV* 0.1 << endl;
 } 
 return 0;
}
```

Question 37

Question:

Write a program to print the first ten natural numbers using while loop statement.

Solution:
```cpp
#include<iostream>
using namespace std;
int main()
{
    int i = 1;
    while (i<=10)
    {
        cout<<"\n "<< i++;
    }
    return 0;
}
```

---

**Question 38**

**Question:** What will be the output of the below program:

```cpp
#include <iostream>
using namespace std;
int main()
{
    int i;
    for (i=1; i<=5; i++)
    {
        if (i==3)
        {
            continue;
        }
        cout<<"\n "<< i;
    }
    return 0;
}
```

---

**Solution:**

1
2
4
5
Question 39

Question:

Write a program to find the size of an array.

Solution:

```cpp
#include <iostream>
using namespace std;

int main()
{
    int num[] = {11, 22, 33, 44, 55, 66};
    int n;

    // Calculating the size of the array with this formula.
    // n = sizeof(array_name) / sizeof(array_name[0])
    // This is a universal formula to find number of elements in
    // an array, which means it will work for arrays of all data
    // types such as int, char, float etc.
    n = sizeof(num) / sizeof(num[0]);
    cout << "Size of the array is:" << n;
    return 0;
}
```

Question 40

Question:

What would be the output of the following programs:

```cpp
#include <iostream>
using namespace std;

int main()
{
    int i;
    for (i=1; i<=5; i++)
    {
        if (i==3)
        {
```
break;
}
cout<<"\n "<< i;
}
return 0;
}

Solution:
1
2

#include<iostream>
using namespace std;
int main()
{
 int i;
for(i=1;i<=5;i++)
{
 if(i==3)
{
 goto HAI;
 }
 cout<<"\n "<< i;
}
HAI : cout<<"\n Linux";
}

Solution:
1
2
Linux

#include<iostream>
using namespace std;
int main()
{
 int i = 54;
 int y = i<<1;
 cout<<"The value of y = "<< y;
 return 0;
}

Solution:
The value of y = 108
```cpp
using namespace std;
int main()
{
    int i = 54;
    int y = i>>1;
    cout << "The value of y = " << y;
    return 0;
}
```

Solution:

The value of y = 27

```cpp
#include<iostream>
#include<cmath>
using namespace std;
int main()
{
    int a, b;
    a = -2;
    b = abs(a);
    cout << "Absolute value = " << b << endl;
    return 0;
}
```

Solution:

Absolute value = 2

```cpp
#include <iostream>
using namespace std;
int main()
{
    for( ; ; )
    {
        cout << "This loop will run forever.\n";
    }
    return 0;
}
```

Solution:

This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever. .........
```cpp
using namespace std;
int main()
{
    cout<<"Hello World!"
    return 0
    cout<<"Hello World!"
}
```

---

**Solution:**

Hello, world!

---

**Question 41**

**Question:**

Write a program to check whether the person is a senior citizen or not.

**Solution:**

```cpp
#include<iostream>
using namespace std;
int main()
{
    int age;
    age=20;
    if(age >= 60)
    {
        cout<<"Senior citizen";
    }
    if(age<60)
    {
        cout<<"Not a senior citizen";
    }
    return 0;
}
```

---

**Question 42**
Question:

Write a program to compute inverse of tan x.

Solution:

```c++
#include<iostream>
#include<math.h>
using namespace std;
int main()
{
    int x = 20;
    cout<<"Inverse of tan x = " << atan(x);
    return 0;
}
```

32 Java Exercises for Beginners

Question 1

Question:

Write a program to print Hello World!.

Solution:

```java
public class MyClass {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

Question 2

Question:

Write a program to find the area of a circle.
Solution:

```java
public class MyClass {
    public static void main(String[] args) {
        int r, area;
        r = 2;
        area = 4 * 3.14 * r * r;
        System.out.println("The area of the circle = " + area);
    }
}
```

Question 3

Question:
Write a program to find the sum of two numbers.

Solution:

```java
public class MyClass {
    public static void main(String[] args) {
        int a, b, sum;
        a = 1;
        b = 2;
        sum = a + b;
        System.out.println("The sum of a and b = " + sum);
    }
}
```

Question 4

Question:
Write a program to find the square of a number.
Solution:

```java
public class MyClass {
    public static void main(String[] args) {
        int a, b;
        a = 2;
        b = a * a;
        System.out.println("The square of a = " + b);
    }
}
```

---

Question 5

**Question:**

Write a program to find the greatest of two numbers.

---

Solution:

```java
public class MyClass {
    public static void main(String[] args) {
        int a, b;
        a = 2;
        b = 3;
        if (a > b) {
            System.out.println("a is greater than b");
        } else {
            System.out.println("b is greater than a");
        }
    }
}
```

---

Question 6

**Question:**
Write a program to print the average of the elements in the array.

Solution:

```java
public class MyClass {
    public static void main(String[] args) {
        int i, avg, sum = 0;
        int [] num = {16, 18, 20, 25, 36};
        for(i=0; i<5; i++)
            sum = sum + num[i];
        avg = sum/5;
        System.out.println("Sum of the Elements in the array = " + sum);
        System.out.println("Average of the Elements in the array = " + avg);
    }
}
```

Question 7

Question:

Write a program such that a Switch (case) allows to make a decision from the number of choices, i.e., from the number of cases.

Solution:

```java
public class MyClass {
    public static void main(String[] args) throws Exception {
        char ch;
        System.out.print("Enter a character:");
        ch = (char)System.in.read();
        switch(ch) {
            case 'R':
                System.out.print("Red");
                break;
            case 'W':
                System.out.print("White");
                break;
            case 'Y':
                System.out.print("Yellow");
                break;
            case 'G':
                System.out.print("Green");
                break;
        }
    }
}
```
Question 8

Question:

Write a program to read 10 numbers from the keyboard and find their sum and average.

Solution:

```java
import java.util.Scanner;
public class MyClass {
    public static void main(String [] args) {
        int N1, N2, N3, N4, N5, N6, N7, N8, N9, N10, sum;
        float X;
        Scanner scan = new Scanner(System.in);
        System.out.println("Enter any ten Numbers: ");
        N1 = scan.nextInt();
        N2 = scan.nextInt();
        N3 = scan.nextInt();
        N4 = scan.nextInt();
        N5 = scan.nextInt();
        N6 = scan.nextInt();
        N7 = scan.nextInt();
        N8 = scan.nextInt();
        N9 = scan.nextInt();
        N10 = scan.nextInt();
        sum = N1 + N2 + N3 + N4 + N5 + N6 + N7 + N8 + N9 + N10;
        X = sum /10;
        System.out.println("The sum of 10 numbers = " + sum);
        System.out.println("The average of 10 numbers = " + X);
    }
}
```
Question 9

Question:

Write a program to print the first 10 numbers starting from one together with their squares and cubes.

Solution:

```java
public class MyClass {
    public static void main(String[] args) throws Exception {
        int i;
        for (i=1; i<=10; i++)
            System.out.println("number = "+i+" its square = "+i*i+" its cube = "+i*i*i);
    }
}
```

Question 10

Question:

Write a program:

If you enter a character M

Output must be: ch = M.

Solution:

```java
public class MyClass {
    public static void main(String[] args) throws Exception {
        char c;
        System.out.print("Enter a character: ");
        c = (char)System.in.read();
        System.out.println("ch= + c");
    }
}
```
Question 11

Question:

Write a program to print the multiplication table of a number.

Solution:

```java
import java.util.Scanner;
public class MyClass {
    public static void main(String[] args) {
        int n, i;
        Scanner scan = new Scanner(System.in);
        System.out.println("Enter a number: ");
        n = scan.nextInt();
        for (i = 1; i <= 5; i++)
            System.out.println(n + " * " + i + " = " + n * i);
    }
}
```

Question 12

Question:

Write a program to print the product of the first 10 digits.

Solution:

```java
public class MyClass {
    public static void main(String[] args) {
        int i, product = 1;
        for (i = 1; i <= 10; i++)
            product = product * i;
        System.out.println("The product of the first 10 digits = " + product);
    }
}
```
Question 13

Question:
Write a program to print whether the given number is positive or negative.

Solution:

```java
public class MyClass {
    public static void main(String[] args) {
        int a;
        a = -35;
        if (a > 0) {
            System.out.println("Number is positive");
        } else {
            System.out.println("Number entered is negative");
        }
    }
}
```

Question 14

Question:
Write a program to check the equivalence of two numbers.

Solution:

```java
import java.util.Scanner;
public class MyClass {
    public static void main(String[] args) {
```
```
int x, y;
Scanner scan = new Scanner(System.in);
System.out.println("Enter a number: ");
x = scan.nextInt();
System.out.println("Enter a number: ");
y = scan.nextInt();
if (x-y==0)
{
    System.out.println("The two numbers are equivalent");
}
else
{
    System.out.println("The two numbers are not equivalent");
}
```

**Question 15**

**Question:**
Write a program to print the remainder of two numbers.

**Solution:**
```
import java.util.Scanner;
public class MyClass {
    public static void main(String [] args) {
        int a, b, c;
        Scanner scan = new Scanner(System.in);
        System.out.println("Enter a number: ");
a = scan.nextInt();
System.out.println("Enter a number: ");
b = scan.nextInt();
c = a%b;
System.out.println("The remainder of a and b = " + c);
    }
}
```
Question:

Write a program to print the given number is even or odd.

Solution:

```java
import java.util.Scanner;
public class MyClass {
    public static void main(String[] args) {
        int a;
        Scanner scan = new Scanner(System.in);
        System.out.println("Enter a number: ");
        a = scan.nextInt();
        if (a % 2 == 0) {
            System.out.println("The number is even");
        } else {
            System.out.println("The number is odd");
        }
    }
}
```

Question 17

Question:

Write a program to print the characters from A to Z.

Solution:

```java
public class MyClass {
    public static void main(String[] args) {
        char a;
        for (a = 'A'; a <= 'Z'; a++)
            System.out.println("\n " + a);
    }
}
```
Question 18

Question:

Write a program to find the incremented and decremented values of two numbers.

Solution:

```java
public class MyClass {
    public static void main(String[] args) {
        int a, b, c, d, e, f;
        a = 10;
        b = 12;
        c = a + 1;
        d = b + 1;
        e = a - 1;
        f = b - 1;
        System.out.println("The incremented value of a = " + c);
        System.out.println("The incremented value of b = " + d);
        System.out.println("The decremented value of a = " + e);
        System.out.println("The decremented value of b = " + f);
    }
}
```

Question 19

Question:

Write a program to calculate the simple interest.

Solution:

```java
public class MyClass {
    public static void main(String[] args) {
        int P, T, R, SI;
        P = 1000;
        T = 2;
        R = 3;
```
\[
SI = \frac{P \times T \times R}{100};
\]

System.out.println("The simple interest = " + SI);

---

**Question 20**

**Question:**

Write a program to Find the largest of three numbers.

**Solution:**

```java
import java.util.Scanner;
public class MyClass {
    public static void main(String [] args) {
        int a, b, c;
        Scanner scan = new Scanner(System.in);
        System.out.println("Enter any number:");
        a = scan.nextInt();
        System.out.println("Enter any number:");
        b = scan.nextInt();
        System.out.println("Enter any number:");
        c = scan.nextInt();
        if(a>b&&a>c)
        {
            System.out.println("a is greater than b and c");
        }
        else if(b>a&&b>c)
        {
            System.out.println("b is greater than a and c");
        }
        else
        {
            System.out.println("c is greater than b and a");
        }
    }
}
```

---

**Question 21**
**Question:**

Write a program to print the factorial of the entered number.

**Solution:**

```java
import java.util.Scanner;

public class MyClass {
    public static void main(String[] args) {
        int i, n, fact = 1;
        Scanner scan = new Scanner(System.in);
        System.out.println("Enter any number:");
        n = scan.nextInt();
        for (i = 1; i <= n; i++) {
            fact = fact * i;
        }
        System.out.println("Factorial of " + n + " is: " + fact);
    }
}
```

---

**Question 22**

**Question:**

Write a program to print the length of the entered string.

**Solution:**

```java
import java.util.Scanner;

public class MyClass {
    public static void main(String[] args) {
        String a;
        Scanner scan = new Scanner(System.in);
        System.out.print("Enter Your Name :");
        a = scan.nextLine();
        System.out.println("The length of the String is: " + a.length());
    }
}
```
Question 23

Question:

Write a program to print the output:

Einstein [0] = E
Einstein [1] = I
Einstein [2] = N
Einstein [3] = S
Einstein [5] = E
Einstein [6] = I
Einstein [7] = N

Solution:

```java
public class MyClass {
    public static void main(String[] args) throws Exception{
        int i;
        char [] num = {'E', 'I', 'N', 'S', 'T', 'E', 'I', 'N'};
        for(i=0; i<8; i++)
            System.out.println("Einstein "+ i + " = " + num[i]);
    }
}
```

Question 24

Question:

Write a program to find square of a number using method.
import java.util.Scanner;
public class MyClass {
    public static void main(String[] args) {
        int x;
        Scanner scan = new Scanner(System.in);
        System.out.println("Enter any number:");
        x = scan.nextInt();
        System.out.println("Square of the number = " + square(x));
    }
    public static int square (int x){
        return x*x;
    }
}

Question 25

Question:
Write a program To print "hello world" 10 times.

Solution:

public class MyClass {
    public static void main(String[] args) {
        int i;
        for (i =1; i<=10; i ++)
            System.out.println("\n hello world");
    }
}

Question 26

Question:
Write a program to print first 5 numbers using do while loop statement.
Solution:

```java
public class MyClass {
public static void main(String[] args) {
    int i = 1;
    do {
        System.out.println("\n" + i++);
    } while (i<=5);
}
}
```

Question 27

Question:

Write a program to print the output:

- body [b] = b
- body [o] = o
- body [d] = d
- body [y] = y

Solution:

```java
public class MyClass {
public static void main(String[] args) throws Exception{
    int i;
    char[] body = {'b', 'o', 'd', 'y'};
    for(i=0; i<4; i++)
        System.out.println("body [" + body [i] + "] - " + body [i]);
}
}
```

Question 28
Question:

Write a program to print the first ten natural numbers using while loop statement.

Solution:

```java
public class MyClass {
    public static void main(String[] args) {
        int i = 1;
        while (i <= 10) {
            System.out.println("\n " + i);
            i++;
        }
    }
}
```

Question 29

Question:

What will be the output of the below program:

```java
public class MyClass {
    public static void main(String[] args) {
        int i;
        for (i = 1; i <= 5; i++) {
            if (i == 3) {
                continue;
            }
            System.out.println("" + i);
        }
    }
}
```

Solution:

```
1
2
4
5
```
Question 30

Question:
Write a program to find the size of an array.

Solution:
```java
public class MyClass {
    public static void main(String[] args) {
        int num [] = {11, 22, 33, 44, 55, 66};
        System.out.println("Size of the array is: " + num.length);
    }
}
```

Question 31

Question:
What would be the output of the following programs:

Solution:

```
public class MyClass {
    public static void main(String[] args) {
        int i;
        for (i=1; i<=5; i++) {
            if (i==3) {
                break;
            }
        }
        System.out.println("" + i);
    }
}
```

1
2
```java
public class MyClass {
    public static void main(String[] args) {
        int x = 2;
        System.out.println(“Square of a number = “ + Math.pow(x, 2));
    }
}
```
Solution:

Enter the name:
Dennis
The name you entered = Dennis

public class MyClass {
    public static void main(String[] args) {
        for ( ; ; )
        {
            System.out.println("This loop will run forever.\n");
        }
    }
}

Solution:

This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever.
This loop will run forever. ........

public class MyClass {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
        System.exit(0);
        System.out.println("Hello, World!");
    }
}

Solution:

Hello,world!

Question 32

Question:

Write a program to check whether the person is a senior citizen or not.
Solution:

```java
public class MyClass {
    public static void main(String[] args) {
        int age;
        age = 20;
        if (age >= 60) {
            System.out.println("senior citizen");
        } else {
            System.out.println("not a senior citizen");
        }
    }
}
```

45 Python Exercises for Beginners

Question 1

Question:

Write a program to Add Two Numbers.

Solution:

```python
a = 1
b = 2
c = a + b
print(c)
```

```python
a = int(input("enter a number: "))
b = int(input("enter a number: "))
c = a + b
print(c)
```

Question 2
Question:

Write a program to find whether a given number (accept from the user) is even or odd, print out an appropriate message to the user.

Solution:

```python
a = int(input("enter a number: "))
if a % 2 == 0:
    print("This is an even number.")
else:
    print("This is an odd number.")
```

Question 3

Question:

Write a program to check whether a number entered by the user is positive, negative or zero.

Solution:

```python
a = int(input("Enter a number: "))
if a > 0:
    print("Positive number")
elif a == 0:
    print("Zero")
else:
    print("Negative number")
```

Question 4

Question:

Write a program to display the calendar of a given date.
Solution:

```python
import calendar
yy = int(input("Enter year: "))
mm = int(input("Enter month: "))
print(calendar.month(yy, mm))
```

Question 5

**Question:**

Write a program to ask the user to enter the string and print that string as output of the program.

Solution:

```python
string = input("Enter string: ")
print("You entered:",string)
```

Question 6

**Question:**

Write a program to Concatenate Two Strings.

Solution:

```python
string1 = input("Enter first string to concatenate: ")
string2 = input("Enter second string to concatenate: ")
string3 = string1 + string2
print("String after concatenation = ",string3)
```
Question 7

Question:
Write a program to Check if an item exists in the list.

Solution:

```python
list_of_items = ["ball", "book", "pencil"]
item = input("Type item to check: ")
if item in list_of_items:
    print("Item exists in the list.")
else:
    print("Item does not exist in the list.")
```

Question 8

Question:
Write a program to Join two or more lists.

Solution:

```python
list1 = ["This", "is", "a", "sample", "program"]
list2 = [10, 2, 45, 3, 5, 7, 8, 10]
finalList = list1 + list2
print(finalList)
```

Question 9

Question:
Write a program to Calculate Cube of a Number.

Solution:

```python
import math
a = int(input("Enter a number: "))
b=math.pow(a,3)
print (b)
```

Question 10

Question:

Write a program to Calculate Square root of a Number.

Solution:

```python
import math
a = int(input("Enter a number: "))
b=math.sqrt(a)
print (b)
```

Question 11

Question:

Write a program that takes a list of numbers (for example, `a = [5, 10, 15, 20, 25]`) and makes a new list of only the first and last elements of the given list.

Solution:

```python
a = [5, 10, 15, 20, 25]
print([a[0], a[4]])
```
Question 12

Question:

Take a list, say for example this one: $a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]$ and write a program that prints out all the elements of the list that are less than 5.

Solution:

```python
a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
for i in a:
    if i < 5:
        print(i)
```

Question 13

Question:

Let's say I give you a list saved in a variable: $a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]$. Write one line of Python that takes this list 'a' and makes a new list that has only the even elements of this list in it.

Solution:

```python
a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
b = [number for number in a if number % 2 == 0]
print(b)
```
**Question 14**

**Question:**

Ask the user for a string and print out whether this string is a palindrome or not (A palindrome is a string that reads the same forwards and backwards).

**Solution:**

```python
a = input("Please enter a word: ")
c = a.casefold()
b = reversed(c)
if list(c) == list(b):
    print("It is palindrome")
else:
    print("It is not palindrome")
```

**Question 15**

**Question:**

Take two lists, say for example these two: `a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]` `b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]` and write a program that returns a list that contains only the elements that are common between the lists (without duplicates). Make sure your program works on two lists of different sizes.

**Solution:**

```python
a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]
result = [i for i in set(a) if i in b]
print(result)
```
**Question 16**

**Question:** Write a program to add a string to text file.

**Solution:**

```python
file = open("testfile.txt", "w")
file.write("Hello World")
file.write("This is our new text file")
file.write("and this is another line.")
file.write("Why? Because we can.")
file.close()
```

**Question 17**

**Question:** Write a program to read a file and display its contents on console.

**Solution:**

```python
with open('testfile.txt') as f:
    line = f.readline()
    while line:
        print(line)
        line = f.readline()
```

**Question 18**
Question:

Take two sets, say for example these two: \( a = \{1, 2, 3, 5, 8, 13, 21, 34, 55, 89\} \) \( b = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13\} \) and write a program that returns a set that contains only the elements that are common between the sets.

Solution:

```python
a = {1, 2, 3, 5, 8, 13, 21, 34, 55, 89}
b = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13}
c = set(a) & set(b)
print(c)
```

Question 19

Question:

Write a program to split the characters of the given string into a list.

Solution:

```python
s = "mystring"
l = list(s)
print(l)
```

Question 20

Question:

Create a program that asks the user for a number and then prints out a list of all the divisors of that number.

Solution:
n=int(input("Enter an integer: ")
print("The divisors of the number are: ")
for i in range(1,n+1):
    if(n%i==0):
        print(i)

Question 21

Question:

Write a program to Find the largest of three numbers.

Solution:

a = int(input("Enter first number: "))
b = int(input("Enter second number: "))
c = int(input("Enter third number: "))
if (a > b) and (a > c):
    largest = a
elif (b > a) and (b > c):
    largest = b
else:
    largest = c
print("The largest number is", largest)

Question 22

Question:

Write a Program to Find Absolute value of a Number.

Solution:

num = int(input("Enter a number: "))
if num >= 0:
    print(num)
else:
Question 23

Question:
Write a program to Find the length of a String.

Solution:

```python
print("Enter 'y' for exit.")
string = input("Enter a string: ")
if string == 'y':
    exit()
else:
    print("Length of the string =", len(string))
```

Question 24

Question:
Write a program to Print Natural Numbers from 1 to N.

Solution:

```python
N = int(input("Please Enter any Number: "))
for i in range(1, N+1):
    print (i)
```
Question:

Write a program to calculate the sum and average of Natural Numbers from 1 to N.

Solution:

```python
N = int(input("Please Enter any Number: "))
sum = 0
for i in range(1,N+1):
    sum = sum + i
print(sum)
average = sum / N
print(average)
```

Question 26

Question:

Write a program to Print a Statement Any Number of Times.

Solution:

```python
n = int(input("Please Enter any Number: "))
for i in range(n):
    print("hello world")
```

Question 27

Question:

Write a program To Multiply Two Numbers Using Function.

Solution:
```python
def my_function():
    a = int(input("enter a number: "))
    b = int(input("enter a number: "))
    c = a * b
    return c
d = my_function()
print(d)
```

---

**Question 28**

**Question:**
Write a program To add an item to the end of the list.

**Solution:**
```python
list1 = ["pen", "book", "ball"]
list1.append("bat")
print(list1)
```

---

**Question 29**

**Question:**
Write a program To remove an item from the list.

**Solution:**
```python
list1 = ["pen", "book", "ball"]
list1.remove("ball")
print(list1)
```
Question 30

Question: Write a program To print the number of elements in an array.

Solution:

```python
list1 = ['pen', 'book', 'ball']
a = len(list1)
print(a)
```

Question 31

Question: Write a program To calculate the variance and standard deviation of the elements of the list.

Solution:

```python
import numpy as np
a = [2, 6, 8, 12, 18, 24, 28, 32]
variance = np.var(a)
std = np.std(a)
print(variance)
print(std)
```

Question 32

Question:
Write a program to get the difference between the two lists.

Solution:

```
list1 = [4, 5, 6, 7]
list2 = [4, 5]
print(list(set(list1) - set(list2))
```

Question 33

Question:

Write a program to select an item randomly from a list.

Solution:

```
import random
list = ['Paper', 'Pencil', 'Book', 'Bag', 'Pen']
print(random.choice(list))
```

Question 34

Question:

Write a program that prints all the numbers from 0 to 6 except 2 and 6.

Solution:

```
for x in range(6):
    if (x == 2 or x==6):
        continue
    print(x)
```
Question 35

**Question:**
Write a program that takes input from the user and displays that input back in upper and lower cases.

**Solution:**
```
a = input("What's your name? ")
print(a.upper())
print(a.lower())
```

Question 36

**Question:**
Write a program to check whether a string starts with specified characters.

**Solution:**
```
string = "myw3schools.com"
print(string.startswith("w3s"))
```

Question 37

**Question:**
Write a program to create the multiplication table (from 1 to 10) of a number.
Solution:

```python
n = int(input("Enter a number: "))
for i in range(1,11):
    print(n, 'x', i, '=' , n*i)
```

Question 38

Question:

Write a program to check a triangle is equilateral, isosceles or scalene.

Solution:

```python
print("Enter lengths of the triangle sides: ")
a = int(input("a: "))
b = int(input("b: "))
c = int(input("c: "))
if a == b == c:
    print("Equilateral triangle")
elif a==b or b==c or c==a:
    print("isosceles triangle")
else:
    print("Scalene triangle")
```

Question 39

Question:

Write a program to sum of two given integers. However, if the sum is between 15 to 20 it will return 20.

Solution:

```python
a = int(input("enter a number: "))
b = int(input("enter a number: "))
```
c = a+b
if c in range(15, 20):
    print (20)
else:
    print(c)

Question 40

Question:
Write a program to convert degree to radian.

Solution:
pi=22/7
degree = int(input("Input degrees: "))
radian = degree*(pi/180)
print(radian)

Question 41

Question:
Write a program to Generate a Random Number.

Solution:
import random
print(random.randint(0,9))

Question 42
Question:

Write a Program to find the semi-perimeter of triangle.

Solution:

```python
a = int(input('Enter first side: '))
b = int(input('Enter second side: '))
c = int(input('Enter third side: '))
s = (a + b + c) / 2
print(s)
```

---

Question 43

Question:

Given a list of numbers, Iterate it and print only those numbers which are divisible of 2.

Solution:

```python
List = [10, 20, 33, 46, 55]
for i in List:
    if (i % 2 == 0):
        print(i)
```

---

Question 44

Question:

Write a program to Multiply all numbers in the list.

Solution:
import numpy
list = [1, 2, 3]
result = numpy.prod(list)
print(result)

Question 45

Question:

Write a program to print ASCII Value of a character.

Solution:

a = 'j'
print("The ASCII value of " + a + " is", ord(a))

Comparison of C, C++ and Java

- C (Successor of BCPL) & C++ (Successor of C) support pointers and structures while Java (Successor of C & C++) do not.
- File generation: C / C++ → .exe files. Java → .class files.
- Translator: C / C++ → Compiler only. Java → (Compiler + interpreter). C / C++ codes are converted into the Machine code after Compilation. But Java codes are First Converted into the Bytes Codes then after it is converted into the Machine Code.
- C uses scanf as input function to read the character or integer entered through the keyboard and printf as output function to print the output on the screen. C++ uses cin as input function to read the character or integer entered through the keyboard and cout as output function to print the output on the screen. But Java uses scan.nextInt() or scan.nextFloat() as input method to read the variable entered through the keyboard and System.out.println as output method to print the output on the screen.

• Pedantically speaking, C / C++ has no "methods" and Java has no "functions". A method in Java can do everything that a function can do in C / C++.

• C & C++ (Compiled languages) are platform dependent whereas Java (Compiled and Interpreted language) is platform independent (Code written in Java can be taken from one computer to the other without having to worry about system configuration details).

• A program written in Java usually requires more memory space than the same program written in C & C++ and it is fast, reliable, and secure. According to Oracle, the company that owns Java, Java runs on 3 billion devices worldwide.

• Java provides both high speed and high performance and reliability, flexibility and seamless integration with other frameworks and technologies — compared to C & C++.

• Java is a popular general-purpose programming language and computing platform that supports multithreading (a process of executing several codes concurrently) while C & C++ do not.

• C is structured language whereas C++ & Java is object oriented language (i.e., C++ & Java has the extensive power and immense extensibility to write large scale complex programs).

• One of the reasons why Java is widely used is because of the availability of huge standard library that consists hundreds of classes and methods under different packages to help software developers. For example:

  java.lang→ for advanced features of strings, arrays etc.
  java.util→ for data structures, regular expressions, date and time functions etc.
  java.io→ for file i/o, exception handling etc.

• Pointers are used in C / C++ program to access the memory and manipulate the address. But Java doesn't support pointers.

• C / C++ Uses header Files but java uses Packages.

• C does not support information hiding. But in C++ & Java, data is hidden by the Encapsulation to ensure that data structures and operators are used as intended.

• Unlike C++, Java & C doesn't support operator overloading.
• There is no Compiler-level support for inheritance in C. But C++ & Java support inheritance. Unlike C++, Java doesn't support multiple inheritance.
• Unlike C / C++, Java doesn't support Unions.
• In C / C++, it is a Programmer responsibility to check the errors. But in Java, it is a system responsibility to check error in the program.
• C / C++ supports preprocessor directives (such as #include, #define) whereas Java doesn't support preprocessor directives.
• C / C++ don’t have built-in support for threads. It relies on third-party libraries for thread support. But Java support threads implicitly.
• C / C++ is nearer to hardware. But Java is not so interactive with hardware.
• C codes can be run by C++ but C cannot run the C++ codes.
• C / C++ support storage classes (like auto, external etc.) whereas Java doesn't support storage classes.
• Unlike Java, C / C++ don’t support Database Connectivity.
• C / C++ uses include statement for including the contents of the screen. But Java uses import statement for including the contents of the screen.
• Unlike C++, C & Java does not support Constructors or Destructors.
• C uses malloc, calloc for storage allocation. C++ uses new, delete for storage allocation whereas Java uses garbage collector for storage allocation.
• C supports 32 keywords. C++ supports 63 keywords. But Java supports 50 defined keywords.
• C & Java does not support templates. But C++ supports templates.
• Unlike C++ & Java, C does not support exception handling.
• C / C++ do not support the use of internet programming method such as Applet. But Java supports the use of applets for the purpose of internet programming.
• C / C++ support Goto keyword and makes use of enum data type. But Java does not support the use of enum data type and Goto keyword.
• C uses top down approach of programming. But C++ & Java uses a bottom up approach of programming.
• C & C++ supports the global variable and the macros concept while Java does not.
The Importance of Algorithms

A Computer Program can be viewed as an elaborate algorithm and algorithms are very important in Computer Science for solving a problem -- based on conducting a sequence of specified actions. The best chosen algorithm usually means a small procedure that solves a recurrent problem and makes sure computer will do the given task at best possible manner. In cases where efficiency matters -- a proper algorithm is really vital to be used. An algorithm is important in optimizing a computer program according to the available resources -- often play a very significant part in the structure of artificial intelligence, where simple algorithms are used in simple applications, while more complex ones help frame strong artificial intelligence.

You might have an algorithm for getting from office to home, for making a chunk of code that calculates the terms of the Fibonacci sequence, or for finding what you’re looking for in a retail store. Algorithms are the building blocks of computer programs or sequence of unambiguous instructions ( the term 'unambiguous' indicates that there is no room for subjective interpretation) that tells how the problem could be addressed and solved -- which is definitely overblown in their importance like road maps for accomplishing a given, well-defined automated reasoning task -- which always have a clear stopping point.

Long division and column addition are examples that everyone is familiar with -- even a simple function for adding two numbers is implementation of a particular algorithm. Online grammar checking uses algorithms. Financial computations use algorithms. Robotic field uses algorithms for controlling their robot using algorithms. An encryption algorithm transforms data according to specified actions to protect it. A search engine like Google uses search engine algorithms (such as, takes search strings of keywords as input, searches its associated database for relevant web pages, and returns results). In fact, it is difficult to think of a task performed by your computer that does not use computer rules that are a lot like a recipes (called algorithms).

The use of computer algorithms (step-by-step techniques used for Problem-solving) plays an essential role in space search programs. Scientists have to use enormous calculations, and they are managed by high-end supercomputers, which are enriched with detailed sets of instructions
that computers follow to arrive at an answer. Algorithms have applications in many different disciplines from science to math to physics and, of course, computing -- and provide us the most ideal option of accomplishing a task. Here is some importance of algorithms in computer programming.

- To improve the effectiveness of a computer program: An algorithm (procedure or formula for solving a problem, based on conducting a sequence of specified actions) can be used to improve the speed at which a program executes a problem and has the potential of reducing the time that a program takes to solve a problem.
- Proper usage of resources: The right selection of an algorithm will ensure that a program consumes the least amount of memory. Apart from memory, the algorithm can determine the amount of processing power that is needed by a program.

The algorithm for a child's morning routine could be the following:

- **Step 1:** Wake up and turn off alarm
- **Step 2:** Get dressed
- **Step 3:** Brush teeth
- **Step 4:** Eat breakfast
- **Step 5:** Go to school

The algorithm to add two numbers entered by user would look something like this:

- **Step 1:** Start
- **Step 2:** Declare variables num1, num2 and sum
- **Step 3:** Read values num1 and num2
- **Step 4:** Add num1 and num2 and assign the result to sum
  \[\text{sum} \leftarrow \text{num1} + \text{num2}\]
- **Step 5:** Display sum
- **Step 6:** Stop

Two of these algorithms accomplish exactly the same goal, but each algorithm does it in completely different way to achieve the required output or to accomplish our task. In computer programming, there are often many different ways – algorithms (any well-defined computational
procedure that takes some value, or set of values, as input and produces some value, or set of values as output) -- to accomplish any given task. Each algorithm has credits and demerits in different situations. If you have a million integer values between -2147483648 and +2147483647 and you need to sort them, the bin sort is the accurate algorithm to use. If you have a million book titles, the quick sort algorithm might be the best choice. By knowing the toughness and weaknesses of the different algorithms, you pick the best one to accomplish a specific task or to solve a specific problem.

One of the most important aspects of an algorithm is how fast it can manipulate data in various ways, such as inserting a new data item, searching for a particular item or sorting an item. It is often easy to come up with a list of rules to follow in order to solve a problem, but if the algorithm is too slow, it's back to the drawing board. Efficiency of an algorithm depends on its design and implementation. Since every procedure or formula for solving a problem based on conducting a sequence of specified actions -- uses computer resources to run -- execution time and internal memory usage are important considerations to analyze an algorithm.

**Why Study Algorithms?**

Algorithms are the heart of computer science (usually means a procedure or basically instance of logic written in software that solves a recurrent problem of finding an item with specific properties among collection of items or transforming data according to specified actions to protect it), and the subject has countless practical applications as well as intellectual depth that is widely used throughout all areas of information technology including solving a mathematical problem (as of finding the greatest common divisor ) in a finite number of steps that often involves repetition of an operation. The word algorithm -- a mathematical concept whose roots date back to 600 AD with invention of the decimal system -- derives from the name of the ninth century Persian mathematician and geographer, Mohammed ibn-Musa al-Khwarizmi, who was part of the royal court in Baghdad and who lived from about 780 to 850. On the other hand, it turns out algorithms (widely recognized as the foundation of modern computer coding) have a long and distinguished history stretching back as far as the Babylonians.
Although there is some available body of facts or information about early multiplication algorithms in Egypt (around 1700-2000 BC) the oldest algorithm is widely recognized to be valid or correct to have been found on a set of Babylonian clay tablets that date to around 1600 - 1800 BC. Their exact significance only came to be revealed or exposed around 1972 when an American computer scientist, mathematician, and professor emeritus at Stanford University Donald E. Knuth published the first English translations of various Babylonian cuneiform mathematical tablets.

Here are some short extracts from his 1972 manuscript that explain these early algorithms:-
"The calculations described in Babylonian tablets are not merely the solutions to specific individual problems; they are actually general procedures for solving a whole class of problems." - Pages 672 to 673 of "Ancient Babylonian Algorithms".

The wedge-shaped marks on clay tablets also seem to have been an early form of instruction manual:-
"Note also the stereotyped ending, 'This is the procedure,' which is commonly found at the end of each section on a table. Thus the Babylonian procedures are genuine algorithms, and we can commend the Babylonians for developing a nice way to explain an algorithm by example as the algorithm itself was being defined...." - Pages 672 to 673 of "Ancient Babylonian Algorithms".

The use of computers, however, has raised the use of algorithms in daily transactions (like accessing an automated teller machine (ATM ), booking an air or train or buying something online) to unprecedented levels of real-world problems with solutions requiring advanced algorithms abounds. From Google search to morning routines, algorithms are ubiquitous in our everyday life -- and their use is only likely to grow to break down tasks into chunks that can be solved through specific implementations. Many of the problems, though they may not seem realistic, need the set of well-defined algorithmic knowledge that comes up every day in the real world. By developing a good understanding of a series of logical steps in an algorithmic language, you will be able to choose the right one for a problem and apply it properly. Different algorithms play different roles in programming – and algorithms are used by computer programs where a program –
• Get input data.
• Process it using the complex logics.
• Stop when it finds an answer or some conditions are met.
• Produce the desired output.

To give you a better picture, here is the most common type of algorithms:

• Searching Algorithms
• Sorting Algorithms
• Path finding Algorithms
• Tree and graph based algorithms
• Approximate Algorithms
• Compression Algorithms
• Random Algorithms
• Pattern Matching
• Sequence Finding and a lot more

You only need to define your problem then select the right algorithm to use. The word algorithm may not appear closely connected to kids, but the truth is that -- for kids -- understanding the process of building a step by step method of solving a problem helps them build a strong foundation in logical thinking and problem solving. Here are some problems you can ask your kid to discuss algorithmic solutions with you:

• How do we know if a number is odd or even?
• How do we calculate all of the factors of a number?
• How can we tell if a number is prime?
• Given a list of ten numbers in random order, how can we put them order?

Algorithms has shown it can yield results in all industries — from predicting insurance sales opportunities and generating the millions of search inquiries every day to automating medicine
research, optimizing transportation routes, and much more. While algorithms help companies like Master Card and Visa to keep their users' information, such as card number, password, and bank statement safely -- algorithms aren't perfect. They fail and some fail spectacularly. Over the past few years, there have been some serious fails with algorithms, which are the formulas or sets of rules used in digital decision-making processes. Now people are questioning whether we're putting too much trust in the algorithms. When algorithms go bad: Online failures show humans are still needed. Disturbing events at Facebook, Instagram and Amazon reveal the importance of context.

The Most Significant Failures When AI Turned Rogue, Causing Disastrous Results

Artificial intelligence (sometimes called machine intelligence) is a part of computer science that emphasizes the creation of intelligent machines with generalized human cognitive abilities that work and reacts like intelligent beings. Artificial intelligence has made a major breakthrough in the processes, including learning (the acquisition of information for using the data), reasoning (using rules to reach definite conclusions) and self-correction -- and advancements are accelerating to present a range of new functionality for businesses. But nothing in this world can be made perfect; hence everything accompanies some notable failures and fallacies in them. Here we list some of the significant AI failures from the last decade that hint that the companies need to work harder and keep coming up with better and improved versions of their innovations.

From self-driving cars to industrial robots, all complex real world problems are being solved with applications of intelligence (AI). Artificial intelligence (AI) is progressing rapidly and makes it possible for machines to think like humans and mimic their actions -- adjust to new inputs and perform human-like tasks by processing large amounts of data and recognizing patterns in the data. While science fiction often renders AI as robots (a machine -- especially one programmable by a computer -- capable of carrying out a complex series of actions without conscious thought or attention) with human-like characteristics, AI can encompass anything from missile guidance to tumor detection to face recognition.
The applications for artificial intelligence are countless and John McCarthy, who coined the term in 1956, defines it as: "the science and engineering of making intelligent machines." The study and design of intelligent agents -- where an intelligent agent is a system that becomes aware or conscious of its environment and takes actions which maximizes its chances of success -- can be applied to many sectors and industries including computer science, psychology, philosophy, neuroscience, cognitive science, linguistics, operations research, economics, control theory, probability, optimization, and logic. The simulation of human intelligence in machines is being tested and used in the maintenance or improvement of health industry for dosing drugs and different treatment in patients, and for surgical procedures in the hospital operating room.

A property of machines: the intelligence that the system demonstrates -- today is properly known as Weak Artificial intelligence, in that it is designed to perform a narrow task (such as web searches, control systems, scheduling, data mining, logistics, speech recognition, facial recognition and many others). However, the long-term goal of many technical researchers is to create Strong Artificial intelligence. While Weak Artificial intelligence may outperform humans at whatever its specific task is, like playing games or solving mathematical problems, Strong Artificial intelligence would outsmart humans at nearly every cognitive task.

In little over a decade, Artificial intelligence (a wide-ranging tool that enables people to rethink how we integrate information, analyze data, and use the resulting insights to improve decision making) has made leaps and bounds. Every single day, a new thousand word post showcase the most recent advancement in Artificial intelligence. Being Artificial intelligence has made remarkable breakthroughs, and many scientists dream of creating the Master Algorithm proposed by Pedro Domingos--which can solve all problems envisioned by humans -- failure is at the core of human advancement-- notable failures are emerging. From self-driving car accidents to Face ID hacks, AI didn't have a perfect year.

The Most Significant Failures When Al Turned Rogue, Causing Disastrous Results:

- **1959:** AI designed to be a General Problem Solver failed to solve real world problems.
- **1982:** Software designed to make discoveries, discovered how to cheat instead.
- **1983**: Nuclear attack early warning system falsely claimed that an attack is taking place.
- **2010**: Complex AI stock trading software caused a trillion dollar flash crash.
- **2011**: E-Assistant told to "call me an ambulance" began to refer to the user as Ambulance.
- **2013**: Object recognition neural networks saw phantom objects in particular noise images.
- **2015**: An automated email reply generator created inappropriate responses, such as writing "I love you" to a business colleague.
- **2015**: A robot for grabbing auto parts grabbed and killed a man.
- **2015**: Image tagging software classified black people as gorillas.
- **2015**: Medical AI classified patients with asthma as having a lower risk of dying of pneumonia.
- **2015**: Adult content filtering software failed to remove inappropriate content, exposing children to violent and sexual content.
- **2016**: AI designed to predict recidivism acted racist.
- **2016**: An AI agent exploited a reward signal to win a game without actually completing the game.
- **2016**: Video game NPCs (non-player characters, or any character that is not controlled by a human player) designed unauthorized super weapons.
- **2016**: AI judged a beauty contest and rated dark-skinned contestants lower.
- **2016**: A mall security robot collided with and injured a child.
- **2016**: The AI "Alpha Go" lost to a human in a world-championship-level game of "Go."
- **2016**: A self-driving car had a deadly accident.
- **2017**: Google Translate shows gender bias in Turkish-English translations.
- **2017**: Facebook chat bots shut down after developing their own language.
- **2017**: Autonomous van in accident on its first day.
- **2017**: Google Allo suggested man in turban emoji as response to a gun emoji.
- **2017**: Face ID beat by a mask.
- **2017**: AI misses the mark with Kentucky Derby predictions.
- **2017**: Google Home Minis spied on their owners.
• 2017: Google Home outage causes near 100% failure rate.
• 2017: Facebook allowed ads to be targeted to "Jew Haters".
• 2018: Chinese billionaire's face identified as jaywalker.
• 2018: Uber self-driving car kills a pedestrian.
• 2018: Amazon AI recruiting tool is gender biased.
• 2018: Google Photo confuses skier and mountain.
• 2018: LG robot Cloi gets stagefright at its unveiling.
• 2018: IBM Watson comes up short in healthcare.

While these are only a few instances of failures that have been observed so far, they are pieces of evidence to the fact that Artificial intelligence (the simulation of human intelligence processes by machines, especially computer systems) has the potential to develop a will of its own that may be in conflict with members of the human race. This is definitely a warning about the potential dangers of Artificial intelligence which should be addressed while exploring its potential interests.

"I believe there is no deep difference between what can be achieved by a biological brain and what can be achieved by a computer. It therefore follows that computers can, in theory, emulate human intelligence — and exceed it."

– Stephen Hawking.

Artificial intelligence in general, context remains a challenge. Despite Its Many Failures, why is artificial intelligence important?

• Artificial intelligence automates repetitive learning and discovery through data.
• Artificial intelligence analyzes more and deeper data.
• Artificial intelligence adds intelligence to existing products.
• Artificial intelligence adapts through progressive learning algorithms to let the data do the programming.
• Artificial intelligence gets the most out of data.
• Artificial intelligence achieves unbelievable accuracy through deep neural networks – which was previously impossible. For example, your interactions with Amazon Alexa, Google Search and Google Photos are all based on deep learning – and they keep getting more precise the more we use them.

The threat of AI-charged job loss is spreading (AI and automation will eliminate the most mundane tasks). No matter what industry you’re in, AI-powered bots (which can answer common questions and point users to FAQs and knowledge base articles) and software are taking a crack at it. Artificial intelligence seems to be ringing the death sound of a bell for all manner of jobs, tasks, chores and activities. From hospitality, to customer service, to home assistants, no job feels safe. Naturally, this has made people worried about the future. But is Artificial intelligence ready to take over our jobs, or even likely to do so ever? Prevalent AI-charged failures would suggest not.

DevOps (a set of software development practices that combines software development (Dev) and information technology operations (Ops) to shorten the systems development life cycle while delivering features, fixes, and updates frequently in close alignment with business objectives) is becoming the standard way of working for Enterprises. Among the few powerful trends we had experienced in the recent times, one is undoubtedly the adoption of DevOps practices – and adoption of DevOps within the organization is rising on a broader scale, and Enterprises are trending toward it. DevOps builds upon best practices to help drive enterprise performance in modernizing environments. It offers organizations a new way to move the business forward and turn technology into a strategic advantage. An increasing number of businesses recognize the power that DevOps can bring a natural extension for Agile and continuous delivery approaches.

"At its essence, DevOps is a culture, a practice, a philosophy."
DevOps expertise is in high demand. Job postings with "DevOps" in a title or keyword are sprouting up everywhere. DevOps is an enterprise software development phrase emerging from combination of IT teams, process and products to enable the continuous delivery of value to end users. It is a firm bond between development and operations that emphasizes a shift in mindset, better collaboration, and tighter integration and aims to create a culture and environment where building, testing, and releasing software can happen rapidly, often, and more reliably, so organizations can solve critical issues quickly, and better serve their customers and compete more effectively in the market.

**What is DevOps?**

"A software development method formed out of a fundamental need that stresses communication, collaboration and integration between software developers and IT professionals." DevOps could be explained simply as operations working together with engineers to get things done faster in an automated and repeatable way.

**History of DevOps**

At the 2008 Agile Toronto conference, Andrew Shafer and Patrick Debois introduced the term in their talk on "Agile Infrastructure". Since 2009, the DevOps term has been steadily promoted based on a simple philosophy — business works best when efforts being coordinated and collaborative — and brought into more mainstream usage through a series of "DevOpsDays", which started in Belgium and has now spread into Web-enabled sphere to resolve the conflict between the software developers and the operations teams when it comes to getting great work done quickly. In recent years, more tangential DevOps initiatives have also evolved, such as OpsDev, WinOps, and BizDevOps to encourage the communication between software developers and IT Operations to increase the speed at which applications being delivered.
Benefits of DevOps

The technical benefits include:

- Continuous software delivery
- Less complexity to manage
- Faster resolution of problems

The cultural benefits include:

- More productive teams
- Higher employee engagement
- Greater professional development opportunities

The business benefits include:

- Faster delivery of features
- More stable operating environments
- Improved communication and collaboration
- More time to innovate (and not fix / keep up)

Features of DevOps

- Source control: Software developers need to safely store their code and keep track of source-code history and versions. For this reason alone, source control is of critical importance.
• **Issue tracking system:** An issue tracking system allows everyone involved to track current issues, estimates, and deadlines.

• **Build system:** The build system supports continuous integration by building the software, running unit and integration tests, deploying to the integration environment, and performing any other automated checks defined for new versions of the software.

• **Monitoring system:** Monitoring systems continuously track all autonomous systems within the DevOps environment, notifying necessary maintenance staff if a system failure occurs.

• **Communications system:** The constant exchange of information is important so email, wiki, and a real-time chat system being enabled for effective communication and collaboration among all members of the project team.

• **Integration environment:** The integration environment hosts all the virtual machines that make up our DevOps environment

• **Code review system:** To make sure software quality, every line of code being reviewed by an experienced developer. The practice of reviewing code also accelerates career growth and learning.

• **Documentation system:** Regrettably, documentation often remains an afterthought in production software projects. To ensure that documentation being written throughout the project, an automated system being developed to allow developers to write documentation easily, along with source code.

**DevOps Goals**

• Improved deployment frequency

• To make faster time to market

• Less failure rate to new releases

• Short lead time between fixes

• Improve mean time to recovery
Is DevOps a good career?

DevOps practitioners are among the highest paid IT professionals today, and the market demand for them is growing rapidly because organizations using DevOps practices are overwhelmingly high-functioning to deliver IT services that offer value to the business. According to a study on the application economy and the role of DevOps, 88% of enterprise IT organizations and LOB (line of business) executives already have planned to adopt DevOps sometime within the next five years to accelerate delivery of apps and offer customers with higher-quality software. In the last two years, listings for DevOps jobs at Indeed.com increased 75 percent. On LinkedIn.com, mentions of DevOps as a skill increased 50 percent. In a recent survey by Puppetlabs, half of their 4,000-plus respondents (in more than 90 countries) said their companies consider DevOps skills when hiring.

Scope of Android Mobile Application Development in India

Android accounts for approximately 85% of all devices sold today. Android Application Development simply means developing new applications that can run on the devices powered by android operating system. Google states that "Android applications can be written using Kotlin (an alternate programming language for developing android applications. Many renowned technology firms have started using Kotlin for developing their android applications such as Pinterest, Uber, Atlassian, Pivotal etc.), Java, and C++ languages" using the Android software development kit, while using other languages is also possible. When thinking about the scope of Android Application Development in India (one of the fastest growing nations in the world as far as IT market is concerned) -- one of the major benefits of choosing android application development is astonishing job opportunities associated with it. Many IT firms and startups require android application developers who can create cost effective apps that are capable of delivering best user experience.

"I think right now it's a battle for the mindshare of developers and for the mindshare of customers, and right now iPhone and Android are winning that battle."

~ Steve Jobs
Android application development is one of the hottest topics in the present time. To be up-to-date with the latest trends in mobile application development, one can perceive by chance or unexpectedly a plethora of tech blogs all over the internet. Contemplating Android application development is a great choice as per current market scenario and importance of Android application development for businesses of today is expanding itself, to wearable, automobiles and other areas. Applications like WhatsApp, Facebook, Twitter, Amazon etc. have brought the world around us in our handset. In a statistical study that spans the America, Europe, Asia, and the Middle East, GlobalWebIndex reports that Android tablets outnumber Apple iPad by more than 34 million and has now garnered the interest of a million smart phone users and it powers hundreds of millions of mobile devices in more than 190 countries of the world.

More than a million applications are available for download at the digital distribution platform operated by Google (double the number of apps that were available in the last few years). And more than 9 million developers write code using Java, XML (the languages that empowers an array of software intended for mobile devices that features an operating system, core applications and middleware). With the increase in the number of Android based smartphones (the devices that we started to use just for the communication purpose (i.e. for talking and messaging), abruptly became the most powerful and dependable source of our day-to-day living) and owing to popularity of android and access of internet over mobiles, people using android smart-phones demand for new Android applications, this in turn creates an outstanding career in technology innovation (to push the boundaries of hardware and software forward to bring new capabilities to users and developers) and a demand for better applications and update for existing one.

The Mobile Application Development is the future of Software Development and Android is on the path of proving the same - according to Google’s Eric Schmidt. Companies like Nokia, BlackBerry, Samsung, HTC, Motorola, Google and many others are going wild with their innovations to alter the software applications according to their requirements to get in touch with millions of users all over the world including their potential customer and the global client base. This adds a big sign of scope for the Android market would be beaming with lots of opportunities in the nearby future.
Introduction

Android is the world's most popular open source mobile operating system (OS) based on the Linux Kernel -- which run on 53 percent of all smartphones in the United States and on 80 percent of all smartphones worldwide -- developed by Android Incorporation (a Palo Alto-based startup company, founded in 2003) and later after acquired by and further advanced by coalition of hardware, software and telecommunications companies i.e., open hand set alliance (a group of 84 technology and mobile companies including Dell, Motorola, Samsung Electronics, Sony, Intel, LG Electronics, Qualcomm, Broadcom, HTC, Sprint, Texas Instruments and Japanese wireless carriers KDDI and NTT DoCoMo etc.) – led by Google -- designed primarily for touchscreen mobile devices such as smartphones and tablet computers. But now this technology is growing at such a rapid pace that it is going to hit the markets of Television, Cars and Wrist Watches very soon too.

Android Architecture

1. Linux Kernel

What is a Kernel? The basic layer is the Linux kernel. The whole Android OS built on top of the Linux Kernel with some further architectural changes made by Google. It is the core part of the Android Operating System that acts as an abstraction layer between the hardware and the rest of the software stack – which consists of drivers (i.e., a well-defined set of instructions – what we call programs or software written in C language that installed into mobile phones and stored in the form of files in the phone) – that tells your mobile phone how to communicate with its hardware components such as camera, display etc. – without which keypad, Bluetooth, Audio, Wi-Fi, Camera won't work properly and it is responsible for:

- **Inter Process Communication** - A Mechanism which allows applications running in different processes to share data and communicate with each other i.e., a mechanism
which allows an application running in a process to send requests and receive responses from an application running in another process.

- **Power Management** (conserves power in the cost of performance and holds the device not to get to sleep state).
- **Memory Management** (make the best or most effective use of memory).

Android uses the Linux Kernel for all its core functionality such as Memory management, process management, networking, security settings etc.

2. **Libraries**

The next layer is the Android's native libraries. It is this layer that enables the device to handle different types of data. These libraries are a Collection of pre-written non-volatile data (written in C / C++ language) and pre-compiled programming codes – which support the well-functioning of android operating system.

Some of the important native libraries include the following:

- Surface Manager / Screen Manager that supports the display screen.
- OpenGL (Open Graphics Library) that supports 3 dimensional graphics.
- SGL (Scalable Graphics Library) that supports 2 dimensional graphics.
- Media Framework that supports recording and playback of audio and video and image formats (MP3, JPG, JPEG, PNG, GIF etc.)
- Free Type that is responsible for font support (i.e., font size, color etc.)
- SSL (Secured Sockets layer) / TLS (Transport Layer Security) that is responsible for internet security and support network applications.
- WebKit that supports the display of web pages (i.e., supports inbuilt browser)
- SQLite that is responsible for storage of user data.
- Bionic is the standard C library that supports embedded Linux-based devices in mobile phones.

3. **Android Run Time (ART)**

Android Runtime consists of Core Java libraries and Dalvik Virtual machine.

- Java Core Libraries that consists of Java packages that enable Android application developers to write Android applications using standard Java programming language.
- DVM (Dalvik Virtual Machine) that is responsible to run android application.

4. **Application Frame Work**

Software Frame work (written in Java language) that supports the features of android applications and manage the basic functions of phone like resource management, voice call management etc.

**Important blocks of Application framework are:**

- **Content Provider** that enable applications to get access data from other applications (such as Contacts), or to share their own data.
- **Notifications Manager** that enables all applications to display custom alerts in the status bar.
- **Activity Manager** that manages the life-cycle of applications and provides a common navigation back stack.
- **Window Manager** that organizes the display screen for the application.
- **Location Manager** that provides the periodic updates of the geographical location of the mobile device using GPS (Global Positioning System which is a satellite-based navigation system) or cell tower.
- **View Manager** that manages the Application User Interface.
- **Package Manager** that provides information about the list of installed apps in Android Mobile Device.
• **Telephony Manager** that provides information about the Telephony Services (such as Phone Network, SIM Serial Number, IMEI Number etc.)
• **XMPP** (Extensible Messaging and Presence Protocol) that supports Online Chat Application (like Yahoo Messenger etc.)
• **Resource Manager** that manages the various types of resources we use in our Application and provides access to non-code resources such as localized strings, graphics, and layout files.

5. Applications

Applications are the top layer in the Android architecture. **Examples of such applications** are:

- SMS client app
- Dialer
- Web browser
- Contact manager
- Facebook
- WhatsApp

**Android Application Development Tools and IDE's:**

- Android SDK (**Software Development Kit**) - It contains debugger, libraries, emulator, sample code, documentation and tutorials.
- Android Studio by Google (official IDE for developing Android Apps)
- **Eclipse IDE** using ADT plugin
- IntelliJ IDEA IDE
- NetBeans IDE

Usage of mobile phones in India has rapidly increased from the past year and counting is still on. Out of the six billion smart phone devices in the world, close to one billion is being used in
India. This comes to about 70% of our current population of India. Lots and lots of startups and other Mobile Application Development industries in India are considering Android Application Development as one of the best remunerative business opportunities. Scope of Android App Development in India is huge since every website or company in India needs its own android app (especially if it is providing a web-based service) to make their business plan into action and for capture their services in phone.

The bright future of the App Development in India can better understand with this one example. The telecommunications companies such as idea, Vodafone, MobikWik, FreeRecharge, Aircel and other cellular depends on the third-party app like, Paytm or free charge for the recharge. Thus they are making their own apps to earn direct profit from it and this is the golden opportunity for the Android Developers. In essence, India considered as a country with several globally recognized IT hubs and Android is a choice at the best for exploration in India.

**Benefits of Choosing Android Application Development:**

a) Android is Open Source  
b) Adaptable User Interface  
c) Massive Mobile App Market
   - Google PlayStore - contains more than 2.7 million android apps  
   - Amazon Appstore - contains 800,000+ apps  
   - Aptoide - contains more than 750,000 apps  
   - Mobile Market - contains more than 800,000 apps  
   - Opera Mobile Store - contains more than 300,000 apps  
   - Mobango-contains over 100k mobile apps  
   - GetJar - contains over 850,000 apps
```c
#include<stdio.h>

int x=1; // implies global variable
main()
{
    int y=3; // implies local variable
}
```

**What will be the output of the following program?**

```java
public class HelloWorld {
    public static void main(String args[]){
        System.out.println(Math.max(1269, 1356));
    }
}
```

**Output on the screen:** 1356

"Whether you want to uncover the secrets of the universe, or you just want to pursue a career in the 21st century, basic computer programming is an essential skill to learn."

− Stephen William Hawking

- Punch Cards (or IBM cards, or Hollerith cards) -- a piece of stiff paper that contained commands for controlling automated machinery or data for data processing applications – was formerly used to program computers.
- Transistors replaced vacuum tubes and ushered in the second generation computer.

**Stuff you need to know about**

1 kilobyte = 1024 bytes

1 megabyte = 1024 \times 1024 bytes
1 gigabyte = 1024 × 1024 × 1024 bytes

- IBM’s first large-scale scientific computer was the IBM 701
- American computer scientist Grace Brewster Murray Hopper completed A-O, a program that allowed a computer user to use English-like words instead of numbers to give the computer instructions. It possessed several features of a modern-day compiler and was written for the UNIVAC I (Universal Automatic Computer I), the first commercial business computer system in the United States.
- ERMA (Electronic Recording Machine, Accounting) was the first machine to use Magnetic Ink Character Recognition (MICR)

Any mathematical expression should be written in C equivalent expression to prevent the display of compilation error on the screen because C language does not accept the general mathematical expressions.

<table>
<thead>
<tr>
<th>Mathematical expression</th>
<th>C equivalent expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x \times \frac{y}{z}$</td>
<td>$x * y / z$</td>
</tr>
<tr>
<td>$(ax + 1) (by + 2)$</td>
<td>$(a * x + 1) * (b * y + 2)$</td>
</tr>
<tr>
<td>$\frac{(a + b)^2}{(a - b)^2}$</td>
<td>$(a+b) * (a+b) / (a-b) * (a-b)$</td>
</tr>
<tr>
<td></td>
<td>or</td>
</tr>
<tr>
<td></td>
<td>pow((a+b), 2) / pow((a - b), 2)</td>
</tr>
<tr>
<td>$\log_{10} \left( \frac{x}{y} + c \right)$</td>
<td>$\log 10 (x/y + c)$</td>
</tr>
<tr>
<td>$ax^2 + bx + c$</td>
<td>$a<em>x</em>x+b*x+c$</td>
</tr>
<tr>
<td>$\ln x$</td>
<td>$\log(x)$</td>
</tr>
<tr>
<td>$\sqrt{p^2 + q^2}$</td>
<td>$\sqrt(p<em>p + q</em>q)$</td>
</tr>
<tr>
<td>$2a^2 + 3b + 2$</td>
<td>$2a*a + 3b + 2$</td>
</tr>
<tr>
<td>$e^x + b$</td>
<td>$\exp (x) + b$</td>
</tr>
<tr>
<td>$\frac{1}{x^2}$</td>
<td>$\sqrt{x}$</td>
</tr>
<tr>
<td>$\frac{1}{x^3}$</td>
<td>$\text{cbrt}(x)$</td>
</tr>
<tr>
<td>$\alpha = \beta + \gamma$</td>
<td>$\text{alpha} = \text{beta} + \text{gamma}$</td>
</tr>
<tr>
<td>$\sin \theta + \cos \theta$</td>
<td>$\sin (\theta) + \cos (\theta)$</td>
</tr>
</tbody>
</table>
\[ a = e^{x/\sqrt{1 + \sin \theta}} \]

- HyperCard (an application program and programming tool for Apple Macintosh and Apple IIGS computers), that was among the first successful hypermedia systems before the World Wide Web

\[
\| \text{ imply or} \\
> \text{ imply greater than} \\
< \text{ imply less than} \\
== \text{ imply equal to} \\
! \text{ imply not} \\
!= \text{ imply not equal to} \\
&& \text{ imply and} \\
& \text{ imply address}
\]

“Measuring programming progress by lines of code is like measuring aircraft building progress by weight.”

– Bill Gates

- American computer scientist John Warner Backus completed Speed code for IBM’s first large-scale scientific computer, the IBM 701. Although using Speed code demanded a significant amount of scarce memory, it greatly reduced the time required to write a program. In 1957, Backus became project leader of the IBM FORTRAN (International Business Machine Formula Translation) project, which became the most popular scientific programming language in history and is still in use today.

- The IBM Harvard Mark I, or the Automatic Sequence Controlled Calculator (ASCC), an electromechanical computer built in 1944 for use by the U.S. Navy Bureau of Ships.

“Programs must be written for people to read, and only incidentally for machines to execute.”

: Harold Abelson
An IBM team led by John Backus developed FORTRAN, a powerful scientific computing language that used English-like statements. Some programmers were skeptical that FORTRAN could be as efficient as hand coding, but that sentiment disappeared when FORTRAN proved it could generate efficient code. Over the ensuing decades, FORTRAN became the most often used language for scientific and technical computing. FORTRAN is still in use today.

<table>
<thead>
<tr>
<th>Header file in C</th>
<th>the functions it defines</th>
</tr>
</thead>
<tbody>
<tr>
<td>stdio.h (standard input output header file)</td>
<td>standard input output functions (like scanf and printf functions)</td>
</tr>
<tr>
<td>math.h</td>
<td>mathematical functions (like log(), sqrt(), sin(), cos(), log10() etc.)</td>
</tr>
<tr>
<td>stdlib.h</td>
<td>standard library functions (like void abort(void) – a function which causes an abnormal/ unusual program termination)</td>
</tr>
<tr>
<td>ctype.h</td>
<td>character manipulation functions (like isalpha() which checks whether a character is an alphabet or not)</td>
</tr>
<tr>
<td>graphics.h</td>
<td>graphical functions</td>
</tr>
<tr>
<td>conio.h (console input output header file)</td>
<td>console input output functions like clrscr() – a function which clears the screen.</td>
</tr>
</tbody>
</table>

“Object oriented programming offers a sustainable way to write spaghetti code. It lets you accrete programs as a series of patches”

- Paul Graham (an English computer scientist, a well-known essayist, programmer, language designer, co-founded Viaweb, invented Bayesian spam filters (basis of modern filters))
“Any fool can write code that a computer can understand. Good programmers write code that humans can understand.”

— Martin Fowler (British software developer, author and international public speaker on software development)

“If you lie to the compiler, it will get its revenge.”

— Henry Spencer (a Canadian computer programmer and space enthusiast. He wrote “regex”, a widely used software library for regular expressions, and co-wrote C News, a Usenet server program)

- Apple engineer William Atkinson designed HyperCard, a software tool that simplified development of in-house applications. In HyperCard, programmers built “stacks” of information with the concept of hypertext links between stacks of pages. As a stack author, a programmer employed various tools to create his own stacks, linked together as a sort of slide show. Apple distributed the program free with Macintosh computers until 1992. HyperCard influenced the creation on the Internet protocol HTTP (Hyper Text Transfer Protocol) and JavaScript.

### Data types and their storage size

<table>
<thead>
<tr>
<th>Data type</th>
<th>Storage size</th>
</tr>
</thead>
<tbody>
<tr>
<td>char</td>
<td>1 byte</td>
</tr>
<tr>
<td>short int</td>
<td>2 byte</td>
</tr>
<tr>
<td>float, long int</td>
<td>4 byte</td>
</tr>
<tr>
<td>double, long double</td>
<td>8 byte</td>
</tr>
</tbody>
</table>

“Most software today is very much like an Egyptian pyramid with millions of bricks piled on top of each other, with no structural integrity, but just done by brute force and thousands of slaves.” — ALAN KAY
• Simula, an object-oriented language, was written by Kristen Nygaard and Ole-John Dahl at the Norwegian Computing Center. Based largely on the Algol 60 programming language, Simula grouped data and instructions into blocks called objects, each represented one facet of a system intended for simulation. In addition to simulation, Simula also had applications in computer graphics, process control, scientific data processing and other fields.

• Pointers are the most beautiful part of C language, but also (ugliest) which brings most trouble to C programmers. Over 90% bugs in the C programs come from pointers.

“C makes it easy to shoot yourself in the foot; C++ makes it harder, but when you do, it blows away your whole leg.”

-- Bjarne Stroustrup

“The best programs are written so that computing machines can perform them quickly and so that human beings can understand them clearly. A programmer is ideally an essayist who works with traditional aesthetic and literary forms as well as mathematical concepts, to communicate the way that an algorithm works and to convince a reader that the results will be correct.”

— Donald Ervin Knuth, Selected Papers on Computer Science

• In 1949, a few years after Von Neumann’s work, the language Short Code appeared (www.byte.com). It was the first computer language for electronic devices and it required the programmer to change its statements into 0’s and 1’s by hand. Still, it was the first step towards the complex languages of today.

“If the code and the comments disagree, then both are probably wrong.”

— Norm Schryer
### C ++ place in the World of Languages

- Ada
- Modula-2
- Pascal
- COBOL
- FORTRAN
- BASIC
- Java
- C#
- C++
- C
- Forth
- Macro-assembler
- Assembler

- C++ was designed to organize the raw power of C using OOP, but maintain the speed of C and be able to run on many different types of computers. C++ is most often used in simulations, such as games. C++ provides an elegant way to track and manipulate hundreds of instances of people in elevators, or armies filled with different types of soldiers. It is the language of choice in today’s AP Computer Science courses.

> “Debugging is twice as hard as writing the code in the first place. Therefore, if you write the code as cleverly as possible, you are, by definition, not smart enough to debug it.”

: **Brian Wilson Kernighan** (a Canadian computer scientist who worked at Bell Labs alongside Unix creators Ken Thompson and Dennis Ritchie and contributed to the development of Unix).

> “Writing code has a place in the human hierarchy worth somewhere above grave robbing and beneath managing.”

: **GERALD WEINBERG**

- Pascal was begun in 1968 by Niklaus Wirth. Its development was mainly out of necessity for a good teaching tool. In the beginning, the language designers had no hopes for it to enjoy widespread adoption. Instead, they
concentrated on developing good tools for teaching such as a debugger and editing system and support for common early microprocessor machines which were in use in teaching institutions.

“Computer science is an empirical discipline. [...] Each new machine that is built is an experiment. Actually constructing the machine poses a question to nature; and we listen for the answer by observing the machine in operation and analyzing it by all analytical and measurement means available. Each new program that is built is an experiment. It poses a question to nature, and its behavior offers clues to an answer.”

− Allen Newell

• An integrated circuit (more often called an IC, microchip, silicon chip, computer chip, or chip) is a piece of specially prepared silicon (or another semiconductor) into which a very complex electronic circuit is etched using photographic techniques.

• The IBM Personal Computer, commonly known as the IBM PC, is the original version and progenitor of the IBM PC compatible hardware platform.

“Code doesn't exist until it's checked into source control.”

: JEFF ATWOOD

• Harvard MBA candidate Dan Bricklin and programmer Bob Frankston developed VisiCalc, the program that turned the personal computer into a business machine.

“Time is so short, you can't make a debug...”

: SCOTT ADAMS

“One of my most productive days was throwing away 1000 lines of code.”

: KEN THOMPSON
• In 1833, Charles Babbage developed the analytical engine. This machine consisted of five functional units such as input unit, memory unit, arithmetic unit, control unit and output unit. The architecture of the modern digital computer resembles the analytical engine and hence Charles Babbage is called the father of computers.

“When I am working on a problem I never think about beauty. I think only how to solve the problem. But when I have finished, if the solution is not beautiful, I know it is wrong.”

— R. BUCKMINSTER FULLER

• C and C++ supports pointers and structures whereas Java does not i.e., Java do not support structures and pointers because JVM (Java virtual machine—a core component of java) do not support structures and pointers.

“Linux is evolution, not intelligent design.”

— Linus Torvalds

“Well, Mr. Frankel, who started this program, began to suffer from the computer disease that anybody who works with computers now knows about. It's a very serious disease and it interferes completely with the work. The trouble with computers is you *play* with them. They are so wonderful. You have these switches - if it's an even number you do this, if it's an odd number you do that - and pretty soon you can do more and more elaborate things if you are clever enough, on one machine.

After a while the whole system broke down. Frankel wasn't paying any attention; he wasn't supervising anybody. The system was going very, very slowly - while he was sitting in a room figuring out how to make one tabulator automatically print arc-tangent X, and then it would start and it would print columns and then bitsi, bitsi, bitsi, and calculate the arc-tangent automatically by integrating as it went along and make a whole table in one operation.

Absolutely useless. We *had* tables of arc-tangents. But if you've ever worked with computers, you understand the disease - the *delight* in being able to see how much you can do. But he got the disease for the first time, the poor fellow who invented the thing.”

— Richard Feynman, Surely You're Joking, Mr. Feynman!: Adventures of a Curious Character
Advantages and disadvantages of using JavaScript

Advantages:

- Supported by so many browsers.
- Simple to learn (its syntax is close to English) and implement.
- Offer a really high speed i.e., instance response to the visitors: you don’t have to wait for a page reload to get your desire result.
- Versatile: JavaScript can also be used inside scripts written in other languages such as Perl and PHP.

Disadvantages:

- Security issue: can be exploited for malicious purposes.
- Cannot be used for any networking applications.

References:

- Let Us C by Yashavant Kanetkar.
- C for Dummies by Dan Gookin.
- C ++: The complete reference by Herbert Schildt.
- Programming with C by Byron S. Gottfried.
- INTRODUCTION to JAVA by Jane Meyerowitz.
- Java 2: The Complete Reference by Herbert Schildt.
- Java For Dummies by Barry Burd.
- C PROGRAMMING NOTE by T K Rajan.
- An Introduction to the C Programming Language and Software Design by Tim Bailey.
- C ++: A Beginners Guide, Teach Yourself C++ by Herbert Schildt.
- C, C ++ & Java (www. w3 schools. com).
- A programming with class: A C++ introduction to computer science by Kamin.
- JAVA hand book by Naughton.
- Teach yourself JAVA by O’Neil.
- AT & T Bell laboratories: The C programmer’s hand book.
- C++ Program Design by Cohoon.
- An introduction to object oriented programming with Java by WU THOMAS.
- Ashok N Kamthane, Programming and Data structures, Pearson Education.
- A TUTORIAL ON POINTERS AND ARRAYS IN C by Ted Jensen.
- Pure basic A beginners guide to Computer Programming by Gary Willoughby.
- C++ for dummies by Stephen Randy Davis.
- C to Java: Converting Pointers into References by Erik D. Demaine.
- Why C++ is not just an Object Oriented Programming Language by Bjarne Stroustrup.
- Java Programming: A Practical Approach by Xavier.
- Herb Schildt's Java Programming Cookbook by Herbert Schildt.
- The Java Programming Language by Arnold.
- Programming in C by Stephen G. Kochan.
- Why learn Linux? By Candace Hazlett
- 11 Reasons Why Linux Is Better Than Windows By Ankush Das
- Linux for Dummies By Richard Blum
- Linux: The Complete Reference By Richard Petersen