1. id command will prints the user id , group id , groups for the current user

   Id

2. id root - this will return userId, groupId and groups for the root by default 0 is reserved for root

   id root

3. **Uid** - It stands for user identifier . The number assigned to each user on the system , identify the user and determine which system resources the user can access.

   uid(0) - this is reserved for root
   uid(1….99) - this is reserved for predefined account
   uid(100 - 999) - these are reserved for system administrator , system accounts / group
   uid (1000-10000) - these are reserved for application account
   uid(above 10000) - user accounts

4. **Gid** - stands for group identifier . The number assigned to each group on the system , identify the group and determine which system resources the group can access.

   gid(0) - this is reserved for root groups
   gid(1-99) - this is reserved for system and application use
   gid(100 and above) - allocated to user groups

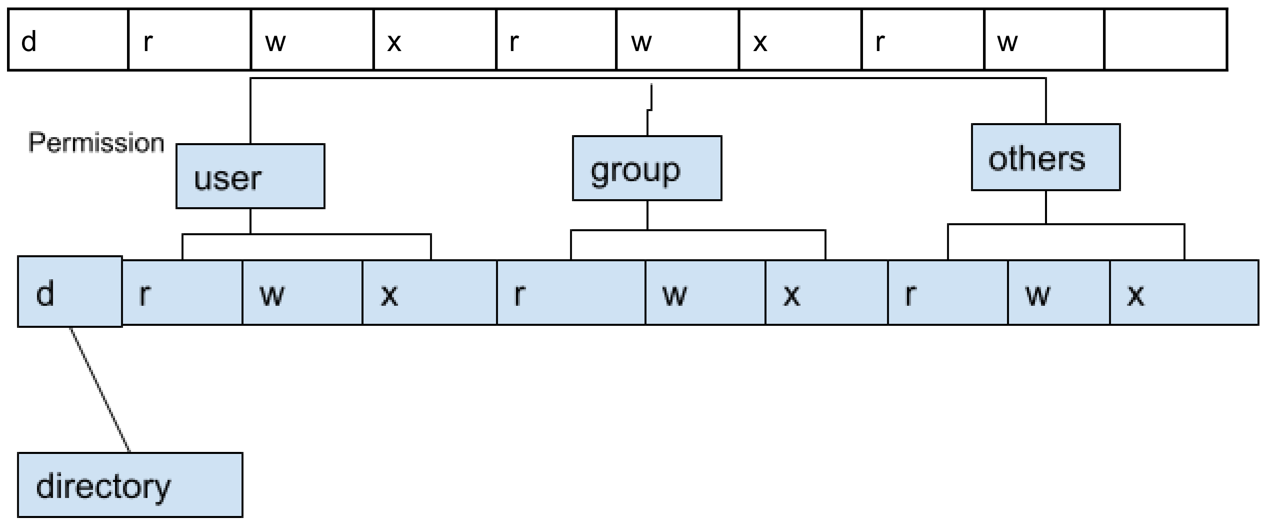
   There are 3 types of permissions that can be provided -

1. Owner
2. Group
3. Others

   Owner permissions are used by the assigned owner of the file/directory . Users belong to this group/class.

   Similarly , group permissions are used by members of the group that own the file or directory. A group is a collection of users . The main purpose of group is to set privileges like read , write , execute to other users

   Other : The permission used by all the users other than file owner , member of the group that owns the file / directory. All the users / groups who do not belong to any class will fall under this class.



7	r	w	x
6	r	w	-
5	r	-	x
4	r	-	-
3	-	w	x
1	-	-	x
0	-	-	-

777	rwx rwx rwx	Read,write,execute permission for all users
755	rwx r-x r-x	Read and execute permission for all the users and file owner/users have permission to write
750	rwx r-x - - -	Read, write , execute permission for users . Read and execute permission for the group and the user who doesn't belong to any group or who is not the owner . don't have access to any file
700	rwx - - - - - -	Only the owner / user of the file has access to

		read,write,and execute the file. Groups and others don't have access to any file.
666	rw- rw- rw-	Read , write permission is given to the owner , group and others. No one is having access to execute the file.
664	rw- rw- r- -	Read , write permission is given to the owner and group. Whereas , read only permission is given to others.
644	rw- r- - r - -	Read and write permission is given to the owner . Read only permission is given to group and others
640	rw- r- - - - -	Read, write permission is given to the owner. Read only permission is given to group and there is no permission given to others.
600	rw- - - - - - -	Only user has the read,write permission , Group and others have no permission.
400	r - - - - - - - -	Owner has the permission to just read. Groups and others have no permission.

Note : Important question for module end exam

groupadd group_name
getent group
usermod -a -G “group_name” “group_name_to_be_added”

chmod 777 test.txt
chmod 400 test.txt

Owner change
chown cdac:check1 test.txt

Group change
Chgrp check1 test.txt

Chgrp vs chown

Chgrp	Chown
chgrp is used to change the ownership of the file	chown will change the ownership of any file / directory.
chgrp is only applicable for group	chown is applicable for both user and group

---

umask:

umask stands for user file creation mask.

We set the default permission of any file / directory to be changed to any specific permission by using umask.

777
543
-
234
W wx r

Shell Scripting

- It is a program to write a series of commands for commands to execute.
- It gathers input from users and executes a program based on the user inputs.
- We can manipulate files and directories
- We can process and manipulate text and files
- It can be held in system administration task such as backup , scheduling any task
- It is also helpful in networking , to ping into any server or download any files .

#!/bin/bash:This specifies the interpreter that we have to execute a script.
#!: this is called as shebang
$:this is shell variable that will hold any variable

```
#!/bin/bash
echo “what's your name”
read name
echo “hi,$name”
```

To find a pattern like “cdac” in a file and once you get the pattern redirect it to new file

```
#!/bin/bash
grep “cdac” filename.txt > out.txt
```

If else statement

```
if [condition]
then
        body
else
        body
fi
```

```
if [condition]
then
        Body
elif [condition]
then
        body
else
        body
fi
```

```
echo "enter your age"
read age
if [ $age -ge 18 ]
then
   echo "Your age is $age and you are eligible"
else
   echo "Your age is $age and you are not eligible"
fi
```

```
#!/bin/bash
echo "enter a number"
read num
if [ $num -gt 0 ]
then
   echo "the number $num is greater"
else
   echo "the number $num is less"
fi
```

```
for in list
do
        body
done

#!/bin/bash
echo "enter number"
read num
for ((i=0;i<=$num;i++))
do
   echo $i
done
```

```
while [ condition ]
do
        body
done
```

```
Case in
        Pattern 1) statement 1 ;;
        Pattern 2) statement 2 ;;
esac
```