## Process Creation

**getpid():** returns the process id of the current process.

**Fork()**

The fork() system call is used to create a new process by duplicating the calling process. The new process created by fork() is called the child process. The original process is called the parent process.

```
#include <sys/types.h>
#include <unistd.h>
#include<stdio.h>
int main(){
printf("Before calling fork\n");
fork();
printf("After calling fork\n");
}
```

```c
#include <sys/types.h>
#include <unistd.h>
#include<stdio.h>
int main(){
printf("Before calling fork\n");
int q=fork();
if(q==0){
printf("Child Process");
}
else
{
printf("Parent process");
}
}
```

```c
#include <sys/types.h>
#include <unistd.h>
#include<stdio.h>
int main(){
printf("Before calling fork\n");
int q=fork();
if(q==0){

printf("child process: My id: %d\n",getpid());
printf("child Process: My_Parent_id:%d\n",getppid());
}
else
{
sleep(1);
printf("Parent Process: My child process id: %d\n",q);
printf("Parent Process: My id:%d\n",getpid());
}
}
```

# Inter Process Communication

**Pipes** are useful for communication between related processes (inter-process communication).

```c
#include <unistd.h>
#include<stdio.h>
int main(){
int fd[2];
int q;
char buf[20];
pipe(fd);
q=fork();
if(q>0){
printf("Parent Process Sending message\n");
write(fd[1],"Hello from Parent \n",20);
}
else{
printf("Child Process receiving \n");
read(fd[0],buf,20);
write(1,buf,20);
printf("Successfully Received\n");
}
}
```