**EXPERIMENT – 6**

**Aim:** Write a program for the implementation of system calls (fork and vfork) of Unix operating systems.

**Description:**

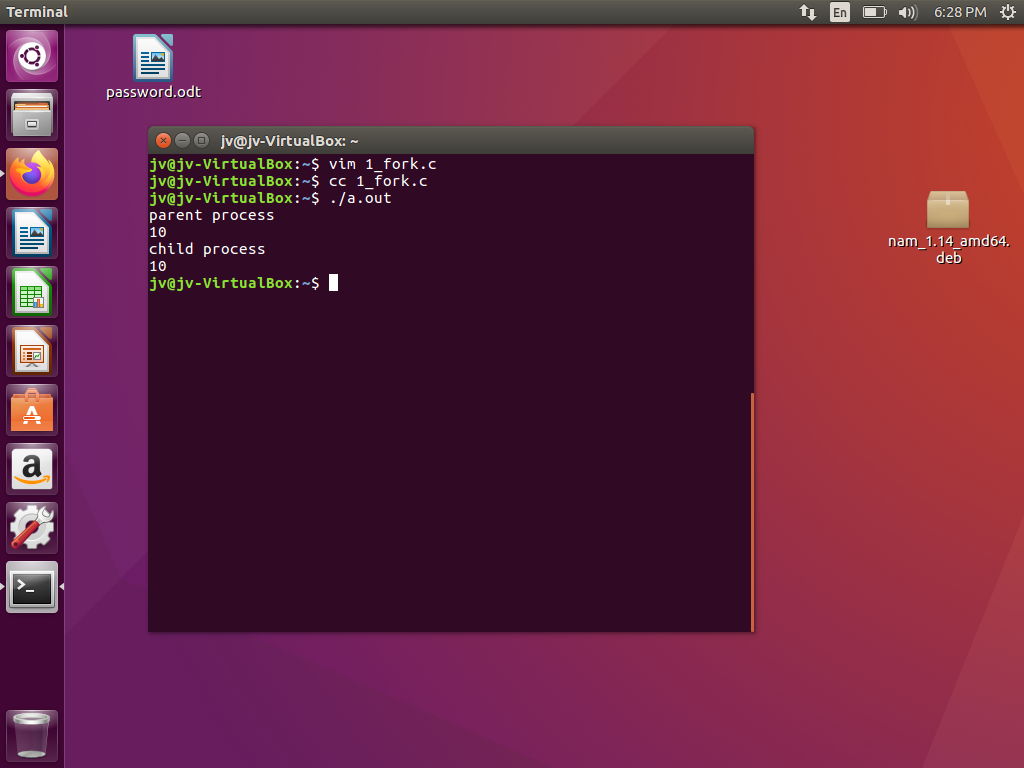
Fork() is a system call that creates a new process. The new process produced by the fork() system call is referred to as the child process, whereas the process that invoked the fork() system call is referred to as the parent process. The code of a child process is identical to that of its parent. Once a child process has been formed, both the parent and child processes begin execution from the next statement following fork(), and both processes are executed at the same time.

Vfork() is another system call that is used to start a new process. The new process produced by the vfork() system call is known as the child process, and the process that invoked the vfork() system call is known as the parent process. The code of the child process is the same as the code of its parent process. Because both processes share the same address space, the child process suspends the parent process's execution until the child process completes its execution.

**Source Code – fork():**

#include<stdio.h>  
#include<unistd.h>  
#include<sys/types.h>  
int main(void)  
{  
int n=10;  
pid\_t pid=fork();  
if(pid==0){  
printf("child process\n");  
}  
else{  
printf("parent process\n");  
}  
printf("%d \n",n);  
return 0;  
}

**Output – fork():**



**Source Code – vfork():**

#include<stdio.h>  
#include<unistd.h>  
#include<sys/types.h>  
int main(void)  
{  
int n=10;  
pid\_t pid=vfork();  
if(pid==0){  
printf("child process\n");  
}  
else{  
printf("parent process\n");  
}  
printf("%d \n",n);  
return 0;  
}

**Output – vfork():**

