**Name: Hamza Rahim**

**Sap-Id: 56776**

**Course: Operating System**

# Lab_Task08

## Installing Essential Build up File:

```
Hamza@Ubuntu:~$ sudo apt install build-essential
[sudo] password for Hamza:
123
Sorry, try again.
[sudo] password for Hamza:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm19
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  bzip2 dpkg-dev fakeroot libalgorithm-diff-perl libalgorithm-diff-xs-perl
  libalgorithm-merge-perl libdpkg-perl libfakeroot libfile-fcntllock-perl
  lto-disabled-list make
Suggested packages:
  bzip2-doc debian-keyring bzr make-doc
The following NEW packages will be installed:
  build-essential bzip2 dpkg-dev fakeroot libalgorithm-diff-perl
  libalgorithm-diff-xs-perl libalgorithm-merge-perl libdpkg-perl libfakeroot
  libfile-fcntllock-perl lto-disabled-list make
0 upgraded, 12 newly installed, 0 to remove and 0 not upgraded.
```

## Creating Directory for Lab008:

```
Hamza@Ubuntu:~$ ls
adir              fstudent.txt  lab_05       lab-5   Music         students
CodeCracker.java  hamza.c       lab05        labo5   Pictures      students.txt
Desktop           hamza.txt     lab08        laboo5  pstudent.txt  Templates
Documents         hello         lab08.c      labs05  Public        Videos
Downloads         hello.c       lab08.class  labs3   RIPHAH
firstprogram.c    hello.cpp     lab08.java   labs4   snap
Hamza@Ubuntu:~$ mkdir lab08
mkdir: cannot create directory ‘lab08’: File exists
Hamza@Ubuntu:~$ mkdir lab008
Hamza@Ubuntu:~$ cd lab008
Hamza@Ubuntu:~/lab008$ touch task01.java
Hamza@Ubuntu:~/lab008$ cat> task01.java
```

## Task01:

```
Hamza@Ubuntu:~/lab008$ touch task01.java
Hamza@Ubuntu:~/lab008$ cat> task01.java
packag org.kodejava.lang.management;
import java.lang.management.ManagementFactory;
import java.lang.management.RuntimeMXBean;
public class GetProcessID{
public static void main(String[] args){
RuntimeMXBean bean=ManagementFactory.getRuntimeMXBean();
String jvmName=bean.getName();
System.out.println("Name="+jvmName):
long pid=long.patseLOng(jvmName.split("@")[0]);
System.out.println("PID="+pid);
}
}Hamza@Ubuntu:~/lab008$ nano task01.java
Hamza@Ubuntu:~/lab008$ java task01.java -o task01
```

## Task01 output:

```
Hamza@Ubuntu:~/lab008$ nano task01.java
Hamza@Ubuntu:~/lab008$ java task01.java -o task01
Name=7184@Ubuntu
PID=7184
```

## Task02:

```
Hamza@Ubuntu:~$ cat>lab08.c
#include<stdio.h>
#include<unistd.h>
int main(){
int p_id,p_pid;
p_id=getpid();
p_pid=getpid();
printf("process ID: %d\n",p_id);
printf("parent Process ID: %d\n",p_pid);
return 0;
}Hamza@Ubuntu:~$ gcc lab08.c -o lab08
Hamza@Ubuntu:~$ ./lab08
process ID: 5718
parent Process ID: 5718
```

## Task03:

```
#include<unistd.h>
#include<stdio.h>
#include<stdlib.h>
int main(void)
{
fork();
int x=5;
pid_t  pid =getpid();
printf("Value of x in PID=%d is %\dn",pid,x);
return 0;
}
```

**Task03 Output:**

```
Hamza@Ubuntu:~/lab008$ gcc task03.c -o task03
Hamza@Ubuntu:~/lab008$ ./task03
Value of x in PID = 8144 is 5
Value of x in PID = 8143 is 5
```

## Task04:

```
Hamza@Ubuntu:~/lab008$ touch task003.c
Hamza@Ubuntu:~/lab008$ cat>task003
#include<unistd.h>
#include<stdio.h>
#include<stdlib.h>
#include<sys/types.h>
int main(void)
{
fork();
pid_t pid=getpid();
int i;
for(i=1;i<=200;i++)
{
printf("This line is from pid%d,valuse=%d\n" ,pid,i);
}
}Hamza@Ubuntu:~/lab008$ gcc task003.c -o task003
```

**Task04 Output:**

```
Hamza@Ubuntu:~/lab008$ gcc task003.c -o task003
Hamza@Ubuntu:~/lab008$ ./task003
This line is from pid9067,valuse=1
This line is from pid9067,valuse=2
This line is from pid9067,valuse=3
This line is from pid9067,valuse=4
This line is from pid9067,valuse=5
This line is from pid9067,valuse=6
This line is from pid9067,valuse=7
This line is from pid9067,valuse=8
This line is from pid9067,valuse=9
This line is from pid9067,valuse=10
This line is from pid9067,valuse=11
This line is from pid9067,valuse=12
This line is from pid9067,valuse=13
This line is from pid9067,valuse=14
```

## Task05:

```c
#include<stdio.h>
#include<sys/types.h>
#include<unistd.h>
void forkexample(){
if(fork()==0)
printf("Hello from the child:\n");
else
printf("Hello From Parents:\n");
}
int main(){
forkexample();
return 0;
}
```

## Task05 output:

```
Hamza@Ubuntu:~/lab008$ nano task04.c
Hamza@Ubuntu:~/lab008$ nano task04.c
Hamza@Ubuntu:~/lab008$ gcc task04.c -o task04
Hamza@Ubuntu:~/lab008$ ./task04
Hello From Parents:
Hello from the child:
```

## Task06:

```c
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <stdlib.h>
#include <errno.h>
#include <sys/wait.h>

int main() {
    pid_t pid;
    int status;

    pid = fork();

    if (pid == -1) {
        printf("Can't fork, error occurred\n");
        exit(EXIT_FAILURE);
    }
    else if (pid == 0) {
        printf("Child process, pid = %u\n", getpid());
        printf("Parent of child process, pid = %u\n", getppid());
```

```c
if (pid == -1) {
    printf("Can't fork, error occurred\n");
    exit(EXIT_FAILURE);
}
else if (pid == 0) {
    printf("Child process, pid = %u\n", getpid());
    printf("Parent of child process, pid = %u\n", getppid());

    char *argv_list[] = {"ls", "-lart", "/home", NULL};
    execv("/bin/ls", argv_list);

    printf("execv failed\n");
    exit(127);
}
else {
    printf("Parent process, pid = %u\n", getpid());
    printf("Parent of parent process, pid = %u\n", getppid());
```

```
else {
    printf("Parent process, pid = %u\n", getpid());
    printf("Parent of parent process, pid = %u\n", getppid());

    if (waitpid(pid, &status, 0) > 0) {
        if (WIFEXITED(status) && !WEXITSTATUS(status))
            printf("Program executed successfully\n");
        else if (WIFEXITED(status) && WEXITSTATUS(status)) {
            if (WEXITSTATUS(status) == 127)
                printf("execv failed\n");
            else
                printf("Program terminated normally but returned a non-zero>
        }
        else
            printf("Program didn't terminate normally\n");
    }
    else {
        printf("waitpid() failed\n");
    }
```

```
        if (waitpid(pid, &status, 0) > 0) {
            if (WIFEXITED(status) && !WEXITSTATUS(status))
                printf("Program executed successfully\n");
            else if (WIFEXITED(status) && WEXITSTATUS(status)) {
                if (WEXITSTATUS(status) == 127)
                    printf("execv failed\n");
                else
                    printf("Program terminated normally but returned a non-zero>
            }
            else
                printf("Program didn't terminate normally\n");
        }
        else {
            printf("waitpid() failed\n");
        }
    }

    return 0;
}

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
```

**Task06 output:**

```
Hamza@Ubuntu:~/lab008$ touch task06.c
Hamza@Ubuntu:~/lab008$ nano task06.c
Hamza@Ubuntu:~/lab008$ gcc task06.c -o task06
Hamza@Ubuntu:~/lab008$ ./task06
Parent process, pid = 9455
Parent of parent process, pid = 6711
Child process, pid = 9456
Parent of child process, pid = 9455
total 12
drwxr-xr-x 23 root  root  4096 Sep  5 08:26 ..
drwxr-xr-x  3 root  root  4096 Sep  5 08:31 .
drwxr-x--- 28 Hamza Hamza 4096 Oct 13 05:05 Hamza
Program executed successfully
```

## Task07:

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/wait.h>

int main() {
    int fd;

    if (fork() != 0) {
        wait((int *)0);
    } else {
        execl("/bin/mkdir", "mkdir", "newdir", (char *)NULL);
        fprintf(stderr, "exec failed!\n");
        exit(1);
    }

    exit(0);
}
```

## Task07 Output:

```
Hamza@Ubuntu:~/lab008$ nano task07.c
Hamza@Ubuntu:~/lab008$ nano task07.c
Hamza@Ubuntu:~/lab008$ gcc task07.c -o task07
Hamza@Ubuntu:~/lab008$ ./task07
```

```
/usr/bin/ld: /usr/lib/gcc/x86_64-linux-gnu/13/../../../x86_64-linux-gnu/Scrt1.o:
 in function `_start':
(.text+0x1b): undefined reference to `main'
collect2: error: ld returned 1 exit status
```