Operating Systems – OS

Lab Task 08 (OS-008)

Submitted by:
Muhammad Roshaan Idrees (56177)

Submitted to:
Sir Shahzad Ahmed Khan

Dated:
14th October 2025

RIPHAH International University

Fall 2025

Faculty of Computing

Tasks

Solution:

Task 1:

```
muhammadroshaanidrees56177@Ubuntu: ~
muhammadroshaanidrees56177@Ubuntu:~$ touch lab08.java
muhammadroshaanidrees56177@Ubuntu:~$ cat<lab08.java
package 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.parseLong(jvmName.split("@")[0]);
        System.out.println("PID = " + pid);
    }
}

muhammadroshaanidrees56177@Ubuntu:~$ java lab08.java -o task1
Name = 4601@Ubuntu
PID = 4601
muhammadroshaanidrees56177@Ubuntu:~$
```

Task 1.2:

```
muhammadroshaanidrees56177@Ubuntu:~$ touch lab08pid.c
muhammadroshaanidrees56177@Ubuntu:~$ cat<lab08pid.c
/*C program to get Process Id and Parent Process Id in Linux.*/

#include <stdio.h>
#include <unistd.h>

int main()
{
    int p_id, p_pid;

    p_id = getpid(); /*process id*/
    p_pid = getpid(); /*parent process id*/

    printf("Process ID: %d\n", p_id);
    printf("Parent Process ID: %d\n", p_pid);

    return 0;
}

muhammadroshaanidrees56177@Ubuntu:~$ gcc lab08pid.c -o task1
muhammadroshaanidrees56177@Ubuntu:~$ ./task1
Process ID: 4707
Parent Process ID: 4707
muhammadroshaanidrees56177@Ubuntu:~$
```

Task 2:

```
muhammadroshaanidrees56177@Ubuntu:~$ touch lab08_e2.c
muhammadroshaanidrees56177@Ubuntu:~$ cat<lab08_e2.c
#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 %d\n", pid, x);
   return 0;
}

muhammadroshaanidrees56177@Ubuntu:~$ gcc lab08_e2.c -o task2
muhammadroshaanidrees56177@Ubuntu:~$ ./task2
Value of X in PID = 4951 is 5
Value of X in PID = 4952 is 5
muhammadroshaanidrees56177@Ubuntu:~$
```

Task 3:

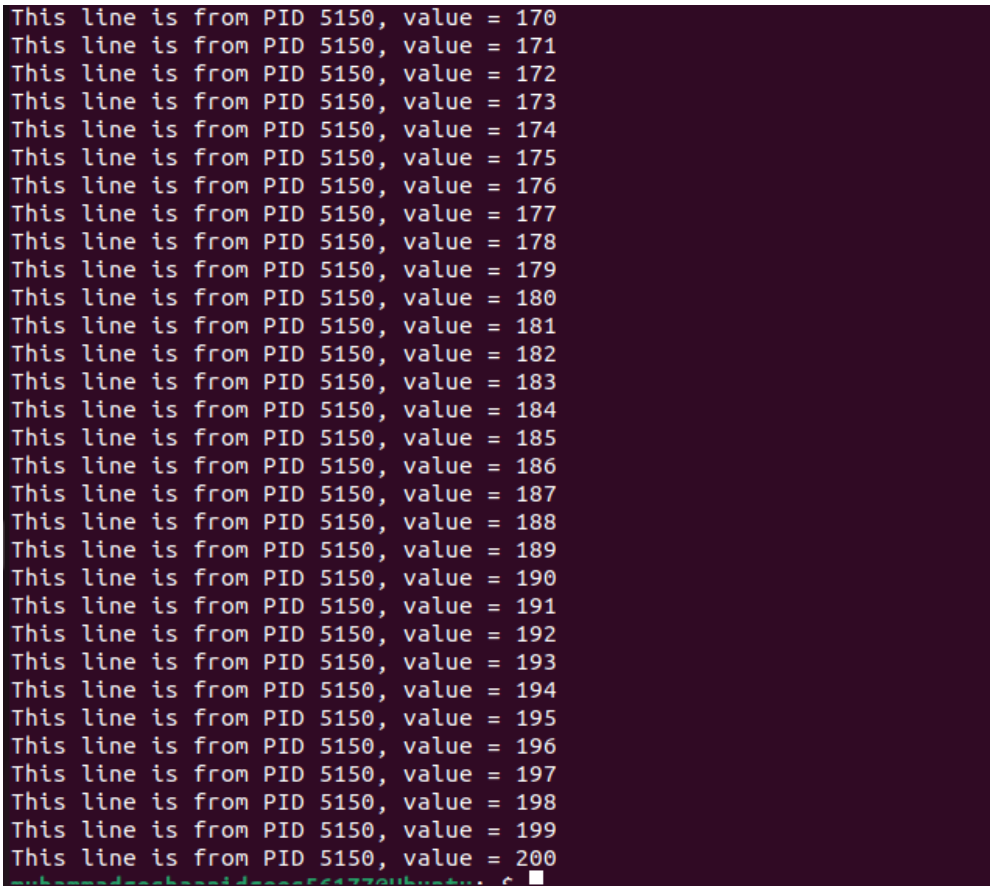
```
muhammadroshaanidrees56177@Ubuntu:~$ touch lab08_e3.c
muhammadroshaanidrees56177@Ubuntu:~$ cat<lab08_e3.c
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>

int main(void)
{
   fork();
   pid_t pid = getpid();
   int i;
   for(i=1; i<=200; i++)
   {
   printf("This line is from PID %d, value = %d\n", pid, i);
   }
}

muhammadroshaanidrees56177@Ubuntu:~$ gcc lab08_e3.c -o task3
muhammadroshaanidrees56177@Ubuntu:~$ ./task3
This line is from PID 5149, value = 1
This line is from PID 5149, value = 2
This line is from PID 5149, value = 3
This line is from PID 5149, value = 4
This line is from PID 5149, value = 5
This line is from PID 5149, value = 6
This line is from PID 5149, value = 7
This line is from PID 5149, value = 8
This line is from PID 5149, value = 9
This line is from PID 5149, value = 10
This line is from PID 5149, value = 11
```

```
This line is from PID 5149, value = 12
This line is from PID 5149, value = 13
This line is from PID 5149, value = 14
This line is from PID 5149, value = 15
This line is from PID 5149, value = 16
This line is from PID 5149, value = 17
This line is from PID 5149, value = 18
This line is from PID 5149, value = 19
This line is from PID 5149, value = 20
This line is from PID 5149, value = 21
This line is from PID 5149, value = 22
This line is from PID 5149, value = 23
This line is from PID 5149, value = 24
This line is from PID 5149, value = 25
This line is from PID 5149, value = 26
This line is from PID 5149, value = 27
This line is from PID 5149, value = 28
This line is from PID 5149, value = 29
This line is from PID 5149, value = 30
This line is from PID 5149, value = 31
This line is from PID 5149, value = 32
This line is from PID 5149, value = 33
This line is from PID 5149, value = 34
This line is from PID 5149, value = 35
This line is from PID 5149, value = 36
This line is from PID 5149, value = 37
This line is from PID 5149, value = 38
This line is from PID 5149, value = 39
This line is from PID 5149, value = 40
This line is from PID 5149, value = 41
This line is from PID 5149, value = 42
```

```
This line is from PID 5149, value = 170
This line is from PID 5149, value = 171
This line is from PID 5149, value = 172
This line is from PID 5149, value = 173
This line is from PID 5149, value = 174
This line is from PID 5149, value = 175
This line is from PID 5149, value = 176
This line is from PID 5149, value = 177
This line is from PID 5149, value = 178
This line is from PID 5149, value = 179
This line is from PID 5149, value = 180
This line is from PID 5149, value = 181
This line is from PID 5149, value = 182
This line is from PID 5149, value = 183
This line is from PID 5149, value = 184
This line is from PID 5149, value = 185
This line is from PID 5149, value = 186
This line is from PID 5149, value = 187
This line is from PID 5149, value = 188
This line is from PID 5149, value = 189
This line is from PID 5149, value = 190
This line is from PID 5149, value = 191
This line is from PID 5149, value = 192
This line is from PID 5149, value = 193
This line is from PID 5149, value = 194
This line is from PID 5149, value = 195
This line is from PID 5149, value = 196
This line is from PID 5149, value = 197
This line is from PID 5149, value = 198
This line is from PID 5149, value = 199
This line is from PID 5149, value = 200
```

```
This line is from PID 5150, value = 1
This line is from PID 5150, value = 2
This line is from PID 5150, value = 3
This line is from PID 5150, value = 4
This line is from PID 5150, value = 5
This line is from PID 5150, value = 6
This line is from PID 5150, value = 7
This line is from PID 5150, value = 8
This line is from PID 5150, value = 9
This line is from PID 5150, value = 10
This line is from PID 5150, value = 11
This line is from PID 5150, value = 12
This line is from PID 5150, value = 13
This line is from PID 5150, value = 14
This line is from PID 5150, value = 15
This line is from PID 5150, value = 16
This line is from PID 5150, value = 17
This line is from PID 5150, value = 18
This line is from PID 5150, value = 19
This line is from PID 5150, value = 20
This line is from PID 5150, value = 21
This line is from PID 5150, value = 22
This line is from PID 5150, value = 23
This line is from PID 5150, value = 24
This line is from PID 5150, value = 25
This line is from PID 5150, value = 26
This line is from PID 5150, value = 27
This line is from PID 5150, value = 28
This line is from PID 5150, value = 29
This line is from PID 5150, value = 30
This line is from PID 5150, value = 31
```

```
This line is from PID 5150, value = 32
This line is from PID 5150, value = 33
This line is from PID 5150, value = 34
This line is from PID 5150, value = 35
This line is from PID 5150, value = 36
This line is from PID 5150, value = 37
This line is from PID 5150, value = 38
This line is from PID 5150, value = 39
This line is from PID 5150, value = 40
This line is from PID 5150, value = 41
This line is from PID 5150, value = 42
```


This line is from PID 5150, value = 170
This line is from PID 5150, value = 171
This line is from PID 5150, value = 172
This line is from PID 5150, value = 173
This line is from PID 5150, value = 174
This line is from PID 5150, value = 175
This line is from PID 5150, value = 176
This line is from PID 5150, value = 177
This line is from PID 5150, value = 178
This line is from PID 5150, value = 179
This line is from PID 5150, value = 180
This line is from PID 5150, value = 181
This line is from PID 5150, value = 182
This line is from PID 5150, value = 183
This line is from PID 5150, value = 184
This line is from PID 5150, value = 185
This line is from PID 5150, value = 186
This line is from PID 5150, value = 187
This line is from PID 5150, value = 188
This line is from PID 5150, value = 189
This line is from PID 5150, value = 190
This line is from PID 5150, value = 191
This line is from PID 5150, value = 192
This line is from PID 5150, value = 193
This line is from PID 5150, value = 194
This line is from PID 5150, value = 195
This line is from PID 5150, value = 196
This line is from PID 5150, value = 197
This line is from PID 5150, value = 198
This line is from PID 5150, value = 199
This line is from PID 5150, value = 200

Task 4:

```
muhammadroshaanidrees56177@Ubuntu:~$ touch lab08_e4.c
muhammadroshaanidrees56177@Ubuntu:~$ cat<lab08_e4.c
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>

void forkexample()
{
    // child process because return value zero
    if (fork() == 0)
        printf("Hello from Child!\n");

    // parent process because return value non-zero.
    else
        printf("Hello from Parent!\n");
}

int main()
{
    forkexample();
    return 0;
}
muhammadroshaanidrees56177@Ubuntu:~$ gcc lab08_e4.c -o task4
muhammadroshaanidrees56177@Ubuntu:~$ ./task4
Hello from Parent!
Hello from Child!
muhammadroshaanidrees56177@Ubuntu:~$
```

Exercise 4:

```
muhammadroshaanidrees56177@Ubuntu:~$ touch lab08_exe4.c
muhammadroshaanidrees56177@Ubuntu:~$ cat<lab08_exe4.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 ret = 1;
    int status;
    pid = fork();

    if (pid == -1){
        // pid == -1 means error occurred
        printf("can't fork, error occurred\n");
        exit(EXIT_FAILURE);
    }
    else if (pid == 0){
        // pid == 0 means child process created
        // getpid() returns process id of calling process
        // Here It will return process id of child process
        printf("child process, pid = %u\n",getpid());
    // Here It will return Parent of child Process means Parent process it self
        printf("parent of child process, pid = %u\n",getppid());

        // the argv list first argument should point to
        // filename associated with file being executed
        // the array pointer must be terminated by NULL
```

```
      // pointer
      char * argv_list[] = {"ls","-lart","/home",NULL};

      // the execv() only return if error occurred.
      // The return value is -1
      execv("/bin/ls",argv_list);
      exit(0);
   }
   else{
      // a positive number is returned for the pid of
      // parent process
      // getppid() returns process id of parent of
      // calling process

      // Here it will return parent of parent process's ID
      printf("Parent of parent process, pid = %u\n",getppid());
      printf("parent process, pid = %u\n",getpid());

      // the parent process calls waitpid() on the child
      // waitpid() system call suspends execution of
      // calling process until a child specified by pid
      // argument has changed state
      // see wait() man page for all the flags or options
      // used here
      if (waitpid(pid, &status, 0) > 0) {
          if (WIFEXITED(status) && !WEXITSTATUS(status))
              printf("program execution successful\n");
          else if (WIFEXITED(status) && WEXITSTATUS(status)) {
              if (WEXITSTATUS(status) == 127) {
                  // execv failed
                  printf("execv failed\n");
```

```
              }
              else
                  printf("program terminated normally, but returned a non-zero status\n");
          }
          else
              printf("program didn't terminate normally\n");
      }
      else {
          // waitpid() failed
          printf("waitpid() failed\n");
      }
      exit(0);
   }
   return 0;
}

muhammadroshaanidrees56177@Ubuntu:~$ gcc lab08_exe4.c -o exer4
muhammadroshaanidrees56177@Ubuntu:~$ ./exer4
Parent of parent process, pid = 3760
parent process, pid = 7238
child process, pid = 7239
parent of child process, pid = 7238
total 12
drwxr-xr-x 21 root                      root                      4096 Sep  9 12:10 ..
drwxr-xr-x  3 root                      root                      4096 Sep  9 21:32 .
drwxr-x--- 23 muhammadroshaanidrees56177 muhammadroshaanidrees56177 4096 Oct 19 13:29 muhammadroshaanidrees56177
program execution successful
```

Task 5:

```
muhammadroshaanidrees56177@Ubuntu:~$ touch lab08_e5.c
muhammadroshaanidrees56177@Ubuntu:~$ cat<lab08_e5.c
#include <stdio.h>
int main()
{
  int fd;
  if (fork() != 0) // for parent
        wait ((int *) 0);
  else  // for child
  {
    excel("/bin/mkdir", "mkdir", "newdir", (char *)NULL);
    fprintf(stderr, "exec failed!\n");
    exit(1);
  }

  exit (0);
}

muhammadroshaanidrees56177@Ubuntu:~$ gcc lab08_e5.c -o task5
lab08_e5.c: In function ‘main’:
lab08_e5.c:5:7: warning: implicit declaration of function ‘fork’ [-Wimplicit-function-declaration]
    5 |   if (fork() != 0) // for parent
      |       ^~~~
lab08_e5.c:6:9: warning: implicit declaration of function ‘wait’ [-Wimplicit-function-declaration]
    6 |         wait ((int *) 0);
      |         ^~~~
lab08_e5.c:9:5: warning: implicit declaration of function ‘excel’ [-Wimplicit-function-declaration]
    9 |     excel("/bin/mkdir", "mkdir", "newdir", (char *)NULL);
      |     ^~~~~
lab08_e5.c:11:5: warning: implicit declaration of function ‘exit’ [-Wimplicit-function-declaration]
   11 |     exit(1);
```

```
muhammadroshaanidrees56177@Ubuntu:~$ touch lab08_e5.c
muhammadroshaanidrees56177@Ubuntu:~$ cat<lab08_e5.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/wait.h>
#include <sys/types.h>

int main()
{
    if (fork() != 0) { // for parent
        wait((int *) 0);
    }
    else { // for child
        execl("/bin/mkdir", "mkdir", "newdir", (char *) NULL);
        fprintf(stderr, "exec failed!\n");
        exit(1);
    }
    exit(0);
}
muhammadroshaanidrees56177@Ubuntu:~$ gcc lab08_e5.c -o task5
muhammadroshaanidrees56177@Ubuntu:~$ ./task5
```
