

Term Work

**On**

**Operating System**

**(PCS 506)**

**Submitted to: Submitted by:**

Dr. Pardeep Singh Anurag Pandey

Assistant Professor University Roll. No.: 2018205

Gehu, Dehradun Class Roll No./Section: 11/A

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**GRTAPHIC ERA HILL UNIVERSITY, DEHRADUN**



## DEPARTMENT OF CSE STUDENT LAB REPORT SHEET

Photograph Passport Size

**Name of Student .................................................... Mob. No ......................................**

**Address Permanent .....................................................................................................**

**Father’s Name ........................... Occupation ...................... Mob. No .........................**

**Mother’s Name ........................... Occupation ...................... Mob. No .......................**

**Section ............ Branch ............ Semester ............ Class Roll No ............ Grade A B C Local Address ................................... Email ............................................ Marks 5 3 1**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S.N**  **o.** | **Practical** | **D.O.P.** | **Date of Submiss ion** | **Grade (Viva)** | **Grade (Report File)** | **Total Marks (out of**  **10)** | **Student’s Signature** | **Teacher’s Signatur e** |
| **1** |  |  |  |  |  |  |  |  |
| **2** |  |  |  |  |  |  |  |  |
| **3** |  |  |  |  |  |  |  |  |
| **4** |  |  |  |  |  |  |  |  |
| **5** |  |  |  |  |  |  |  |  |
| **6** |  |  |  |  |  |  |  |  |
| **7** |  |  |  |  |  |  |  |  |
| **8** |  |  |  |  |  |  |  |  |
| **9** |  |  |  |  |  |  |  |  |
| **10** |  |  |  |  |  |  |  |  |
| **11** |  |  |  |  |  |  |  |  |
| **12** |  |  |  |  |  |  |  |  |

# PRACTICAL 1

**Question:** Write a C program to demonstrate the use of fork() system call.

**About Fork() function:**

Fork system call is used to create new process which is called child process which runs concurrently with the parent process. Parent process is the process which makes the fork() call. Fork() function is defined in header unistd.

Fork() system call is Unix/Linux specific system call. PID is Process Identification Number on Linux/Unix OS.

## Source Code:

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

int main()

{

printf("Name: Anurag Pandey \nSection: A \nStudent ID: 20011436\n"); fork();

printf("Hello World!\n"); printf("PID: %d\n", getpid());

return 0;

}

**Output:**

# PRACTICAL 2

**Question:** Write a C program in which parent process computes the sum of even Numbers and child process computes the sum of odd number stored in an array using a fork().

First the child process should print its answer i.e sum of odd number then the parent process should print its answer i.e the sum of even number.

**PID:** PID is Process Identification Number on Linux/Unix OS. In child process, it returns 0

**Source Code:**

#include<stdio.h> #include<unistd.h> int main()

{

printf("Name: Anurag Pandey\nSection: A \nStudent ID: 20011436n"); int even\_sum = 0, odd\_sum = 0, n;

printf("Enter size of array: "); scanf("%d",&n);

int arr[n];

printf("Enter numbers:\n"); for(int i = 0; i < n; i++)

{

scanf("%d",&arr[i]);

}

int pid = fork(); if(pid != 0)

{

for(int i = 0; i < n; i++)

{

if(arr[i]%2 != 0)

odd\_sum += arr[i];

}

printf("Sum of Odd Numbers: %d\n", odd\_sum);

}

else

{

for(int i = 0; i < n; i++)

{

if(arr[i]%2 == 0)

even\_sum += arr[i];

}

printf("Sum of Even Numbers: %d\n",even\_sum);

}

return 0;

}

**Output:**

# PRACTICAL 3

**Question:** Write a C program to demonstrate Orphan Process using fork function.

**Source Code:**

#include<stdio.h> #include<unistd.h> #include<stdlib.h> int main()

{

int pid = fork(); if(pid!=0)

{

printf("parent process \n "); exit(0);

}

else if (pid==0){

sleep(2);

printf("child process \n");

}

return 0;

}

**Output:**

# PRACTICAL 4

**Question:** Write a C program to demonstrate Zombie Process using fork function.

**Source Code:**

#include<stdio.h> #include<unistd.h> #include<stdlib.h> int main()

{

int pid = fork(); if(pid!=0)

{

}

else

{

}

sleep(2);

printf("parent process \n ");

printf("child process \n"); exit(0);

return 0;

}

## Output: