Assignment – 3

1.Which signals are triggered, when the following actions are performed.

1. **user press ctrl+C**
2. **kill() system call is invoked**
3. **CPU tried to execute an illegal instruction**
4. **When the program access the unassigned memory**

Soln:-

1. SIGINT (Signal Interrupt)
2. SIGKILL (Signal Kill)
3. SIGILL (Signal Illegal Instruction)
4. SIGBUS

2. List the gdb command for the following operations

1. **To run the current executable file**[1]
2. **To create breakpoints at**
3. **To resume execution once after breakpoint**
4. **To clear break point created for a function**
5. **Print the parameters of the function in the backtrace**

Soln:-
1. run
2. break
3. continue
4. clear
5. print

3. Guess the output of the following program

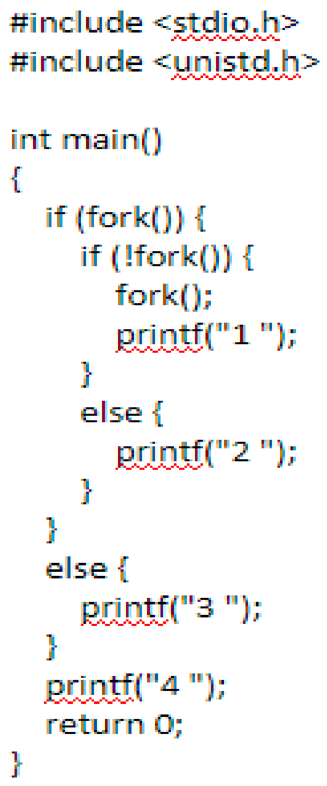
```
#include<stdio.h>
#include<unistd.h>
int main()
{ if(fork()&&(!fork())){
    if(fork()||fork()){
     fork();
     }
   }
 printf("2");
 return 0;
}
```

1 Ponkumaran T, Mepco Schlenk Engineering College, Sivakasi

Soln:-

2222222

4.Guess the output for the following program.



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

int main()
{
    if (fork()) {
        if (!fork()) {
            fork();
            printf("1 ");
        }
        else {
            printf("2 ");
        }
    }
    else {
        printf("3 ");
    }
    printf("4 ");
    return 0;
}
```

Soln:

243414

5.Create two thread functions to print hello and world separately and create threads for each and execute them one after other in C

Soln:-

```
#include <stdio.h>
#include <pthread.h>


void *printHello(void *arg) {
   printf("Hello ");
   pthread_exit(NULL);
}


void *printWorld(void *arg) {
   printf("World\n");[2]
   pthread_exit(NULL);
}

int main() {
   pthread_t thread1, thread2;


   if (pthread_create(&thread1, NULL, printHello, NULL) != 0) {
       fprintf(stderr, "Error creating thread 1\n");
       return 1;
   }


   pthread_join(thread1, NULL);


   if (pthread_create(&thread2, NULL, printWorld, NULL) != 0) {
       fprintf(stderr, "Error creating thread 2\n");
       return 2;
   }


   pthread_join(thread2, NULL);

   return 0;
}
```

6. How to avoid Race conditions and deadlocks?

Soln:-

When two thread try to access the same data at a same time the deadlock occurs, however these deadlock can be avoided by the synchronization mechanisms, deadlock detection and recovery and atomic operations.

7. What is the difference between exec and fork ?

Soln:-

Fork()

It is a system call in the C Programming Language.
It is used to create a new processes
Its return value is an integer type
It does not take any parameters
It can return three type of integer values

2 Ponkumaran T, Mepco Schlenk Engineering College, Sivakasi

Exec()
It is a system call of operating system
exec() runs an executablefile
It does not creates new processes
Process Identifier does not change.
In exec() machine code, data heap and stack of the process are replaced by a new program.

8. What is the difference between process and threads.

Soln:-[3]

A process is a independent program in execution and it has its own memory space, resources, and state wherease a thread is a lightweight unit in execution of the process. Multiple Thread within processes share the same resources, including memory.

9.Write a C program to demonstrate the use of Mutexes in threads synchronization...explain it and give me one sample program

Soln:-

```
#include <stdio.h>
#include <pthread.h>

#define NUM_THREADS 5

int sharedCounter = 0;
pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;

void *incrementCounter(void *threadID) {
    long tid = (long)threadID;


    pthread_mutex_lock(&mutex);


    sharedCounter++;
    printf("Thread %ld incremented the counter. Current value: %d\n", tid, sharedCounter);


    pthread_mutex_unlock(&mutex);

    pthread_exit(NULL);
}

int main() {
    pthread_t threads[NUM_THREADS];
    int rc;
```

3 Ponkumaran T, Mepco Schlenk Engineering College, Sivakasi

```
    long t;

    for (t = 0; t < NUM_THREADS; t++) {
        printf("Creating Thread %ld\n", t);
        rc = pthread_create(&threads[t], NULL, incrementCounter, (void *)t);

        if (rc) {
            printf("Error: Return code from pthread_create() is %d\n", rc);
            return 1;
        }
    }


    for (t = 0; t < NUM_THREADS; t++) {
        pthread_join(threads[t], NULL);
    }

    printf("All threads have completed. Final counter value: %d\n", sharedCounter);

    return 0;
}
```
[4]

4 Ponkumaran T, Mepco Schlenk Engineering College, Sivakasi