Assignment 1

Course Instructor

Asif Muhammad

Submitted by:

Haroon Aziz(22i2697)

Date

5th Oct, 2025

Fall 2025

Department of Software Engineering

FAST – National University of Computer & Emerging Sciences

Question 1:

Q1 > C l222697_Q1.c

```c
#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;
}
```

Dry Run:

Step 1 (Fork A):

P0 (parent) gets >0 > condition true > goes inside first if.

C1 (child) gets 0 > condition false > goes to else > prints: 3

Now processes: P0, C1

Step 2 (Fork B, only in P0 path):

P0 runs if (!fork()):

In P0 (parent): fork()>0 > !true = false > else > prints: 2

New child C2: fork()==0 > !false = true > enters true branch

Step 3 (Fork C, only in C2):

C2 calls fork() > creates C3

Both C2 and C3 print: 1

Step 4 (Final print):

printf("4 ") runs in everyone: P0, C1, C2, C3

So four 4's total

2) Process Hierarchy (who prints what)

P0 > prints: 2, 4

C1 (from Fork A) > prints: 3, 4

C2 (from Fork B) > prints: 1, 4

C3 (from Fork C) > prints: 1, 4

Total processes: 4 (P0, C1, C2, C3)3)

Sample Output (order varies)3 4 2 4 1 4 1

Breakdown:

3 > C1 (else of first if)

2 > P0 (else after !fork is false in parent)

1 > C2 and C3 (true branch work)

4 > all four processes at the end

Question 2:

```c
Q2 > C l222697_Q2.c
#include <stdio.h>
#include <unistd.h>

int main()
{
    if (fork() && (!fork())) {
        if (fork() || fork()) {
            fork();
        }
    }
    printf("2 ");
    return 0;
}
Ctrl+I to chat, Ctrl+K to generate
```

Dry Run:

First fork() creates P1.

P1 gets 0 > left side of && is false > skips block > prints 2.

P0 continues to the second fork.

Second fork() runs in P0.

P0: !fork() = false > skips inner block > prints 2.

P2 (child): !fork() = true > enters the nested if.

In P2: executes (fork() || fork()).

The first inner fork creates P3.

- P2 (parent): first fork returns PID > condition true > skips second fork > body fork creates P6.
- P3 (child): first fork = 0 > second fork runs > creates P4.
- P3 (parent of second fork): condition true > body fork creates P5.
- P4 (child): both forks returned 0 > condition false > skips body.

Total processes: 7 > P0, P1, P2, P3, P4, P5, P6.

Each process prints "2 " once.

Output :

2 2 2 2 2 2 2

Question 3:

Q3 > C I222697_Q3.c

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

int main()
{
    fork();
    fork() && fork() || fork();
    fork();
    printf("forked\n");
    return 0;
}
```

Dry Run:

StepbyStep

After F1: 2 processes.

F2 (E = A && B || C)

Parent path (A true):

Run B.

B parent → (A&&B) true → skip C.

B child → (A&&B) false → run C.
Total from parent side: 3.

Child path (A false):
Skip B, run C.
Total from child side: 2.

After F3:
Every process forks once → ×2.

Count Summary
Start: 1
After F1: 2
After F2: 10 (2×5)
After F3: 20

Output

Each prints “forked” → 20 lines.