

Worksheet 3

1/22/2025

6 Points Possible

Attempt 1



1/21/2025

NEXT UP: Review Feedback

Attempt 1 Score:

N/A



Add Comment

Unlimited Attempts Allowed

Details

Q1: What would be the output of the following program?

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
int main()
{
    // make two process which run same
    // program after this instruction
    fork();

    printf("Hello world!\n");
    return 0;
}
```

Q2: What would be the output of the following program?

```
#include <stdio.h>
#include <sys/types.h>
int main()
{
    fork();
    fork();
    fork();
    printf("hello\n");
    return 0;
}
```

Q3: Where would you place the code for exec to run it in the child process?

```

/* fork a child process */
pid = fork();
if (pid < 0) { /* error occurred */
    fprintf(stderr, "Fork Failed");
}
else if (pid == 0) {
    /* code 1 */
}
else {
    /* code 2 */
}

```

Answer 1:


fork() creates a child process, so it will print Hello World! just like the main program does. The next Hello World! will be in the next line instead of side by side as there is \n too after Hello World!

Output:

Hello World!

Hello World!

Answer 2:

for each fork, there will be two new child processes, so,
so, for n fork() calls, there will be  2^n processes.

1st fork() : 1 -> 2 processes

2nd fork() : 2 -> 4 processes

3rd fork() : 3 -> 8 processes

so, hello will be printed 8 times and due to the \n, they will be one after another in the new line.

Output:

hello

hello

hello

hello

hello

hello

hello

hello

Answer 3:

fork() function gives a value which is put into the pid variable here.

reference: pid = fork()

fork() can have three types of values, negative, zero, and positive.

negative: fork failed, no child process was found

zero: child process was created

positive: returned to parent process, not child

exec() needs to be in the "**code1**" block (when pid == 0) as we want the exce() to run in the "**child process**".

[New Attempt](#)