Operating Systems Ex2

1.
Assuming all the fork() calls succeeded:

The father process is waiting for the child.

The child returns as the exit status “3”.

The father is taking that exit status, and adding 1 to it, so now value=4.

The father process prints **4** and returns 4 from main().

2.
Although the child process is changing the global variable, it is different than the global variable that the father has. So the father process will print: **“Parent: value = 5”**.

3.
Four new processes.

	Line 1	Line 2	Line 3	Line 4	Line 5
father	a = 0	Creates child1. a = 2	Creates child2. a = 2	a = 3	Creates child3.
child1	-	Created by father. a = 3	Skips the if	a = 4	Skips the if
child2	-	-	Created by father. a = 2	a = 3	Creates child4.
child3	-	-	-	-	Created by father
child4	-	-	-	-	Created by child2.

4.
127.
Every “fork()” call doubles the number of processes running. So let’s count the total number of those calls:

The first line of the loop is being called 5 times, and the second line is being called 2 more.

So in total there are 7 calls, meaning there are 128 processes in the end. The father process is not being created, so there are a total of 128-1 = 127 processes.