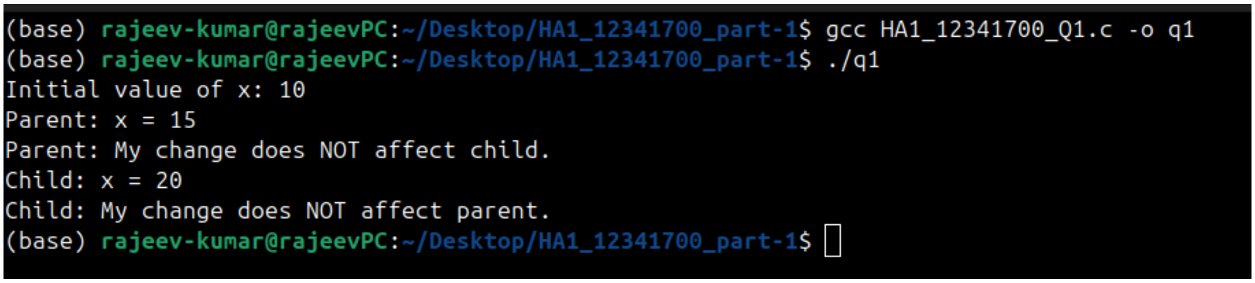
Part 1 - FORK

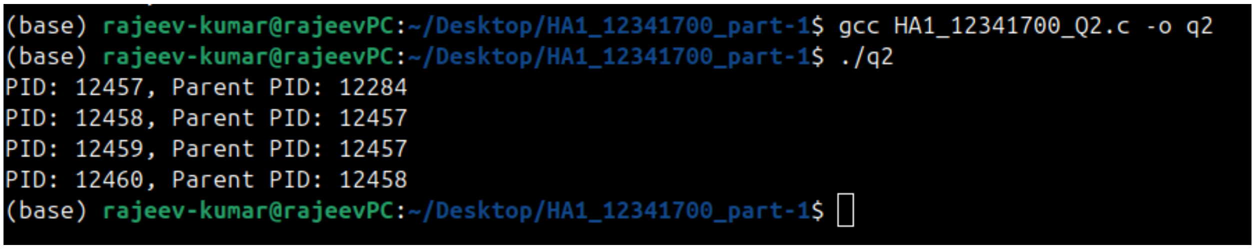
Question 1 :



```
(base) rajeev-kumar@rajeevPC:~/Desktop/HA1_12341700_part-1$ gcc HA1_12341700_Q1.c -o q1
(base) rajeev-kumar@rajeevPC:~/Desktop/HA1_12341700_part-1$ ./q1
Initial value of x: 10
Parent: x = 15
Parent: My change does NOT affect child.
Child: x = 20
Child: My change does NOT affect parent.
(base) rajeev-kumar@rajeevPC:~/Desktop/HA1_12341700_part-1$
```

- When `fork()` is called, the entire process memory is duplicated (including global variables, heap, stack).
- The child gets its own copy of x initialized to the same value (10).
- Changes in the child's memory do not affect the parent's memory because after fork(), they have separate address spaces.
- This behavior is called Copy-on-Write in modern OS — actual copying happens only when a write occurs.

Question 2 :



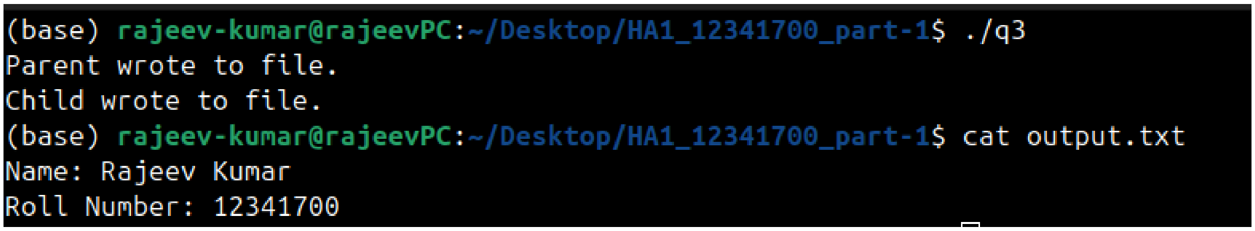
```
(base) rajeev-kumar@rajeevPC:~/Desktop/HA1_12341700_part-1$ gcc HA1_12341700_Q2.c -o q2
(base) rajeev-kumar@rajeevPC:~/Desktop/HA1_12341700_part-1$ ./q2
PID: 12457, Parent PID: 12284
PID: 12458, Parent PID: 12457
PID: 12459, Parent PID: 12457
PID: 12460, Parent PID: 12458
(base) rajeev-kumar@rajeevPC:~/Desktop/HA1_12341700_part-1$
```

- First fork(): Creates 2 processes.
- Second fork(): Each of those 2 processes executes the second fork, resulting in 4 processes total.
- Process tree:

```
  P0
 /   \
C1   C3
|
C2
```

- Total = 2^n processes for n forks

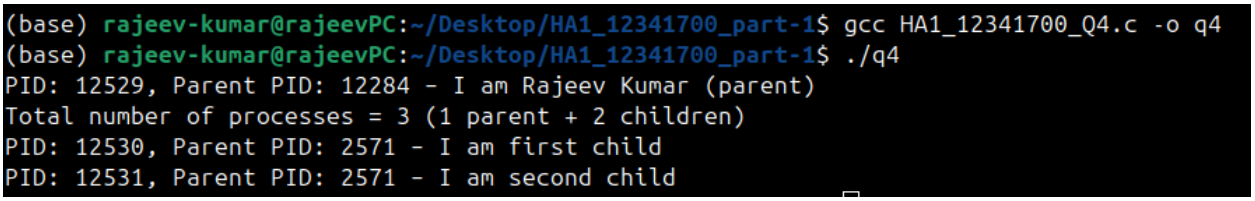
Question 3 :



```
(base) rajeev-kumar@rajeevPC:~/Desktop/HA1_12341700_part-1$ ./q3
Parent wrote to file.
Child wrote to file.
(base) rajeev-kumar@rajeevPC:~/Desktop/HA1_12341700_part-1$ cat output.txt
Name: Rajeev Kumar
Roll Number: 12341700
```

- File descriptors are inherited after fork — both processes share the same open file description (file offset + flags).
- If both write without synchronization, output order can be interleaved depending on scheduling.
- Since the file offset is shared, writes append sequentially (unless manually repositioned).

Question 4 :



```
(base) rajeev-kumar@rajeevPC:~/Desktop/HA1_12341700_part-1$ gcc HA1_12341700_Q4.c -o q4
(base) rajeev-kumar@rajeevPC:~/Desktop/HA1_12341700_part-1$ ./q4
PID: 12529, Parent PID: 12284 - I am Rajeev Kumar (parent)
Total number of processes = 3 (1 parent + 2 children)
PID: 12530, Parent PID: 2571 - I am first child
PID: 12531, Parent PID: 2571 - I am second child
```

- The parent first forks the first child. Then, still in the parent branch, it forks the second child.
- Structure:

```
     Parent
      /    \
    C1     C2
```

- The total number of processes = 3 (parent + 2 children).
- PIDs reflect hierarchy: parent’s PID is the PPID of each child.