OS Assignment 3 - Q2

- The program uses a random string generator to build an array of 50 random strings of length 7 and sends a burst of 5 strings in the format: `(index0)(string0)(index1)(string1)...(index4)(string4)`
- For each IPC mechanism, P1 sends the strings and P2 receives and sends the highest index back to P1
- This functionality is achieved by calling `fork()` in P1 in order to execute P2 by calling `execl()`

2.1 Unix Domain Sockets

```
P1: socket() → bind() → listen() → accept() → send() & recv() →
close()
P2: socket() → connect() → send() & recv() → close()
```

P1 creates the socket and binds to the specified address (“`csock`”). It then calls listen() and accept() to wait for P2 to call connect() to connect to the socket. P1 then calls `send()` on the burst of 5 strings; P2 calls `recv()` to read the 5 strings, extracts the highest index from the entire string, and sends back a string of length 3 - `(highest index)(~)`. P1 calls `recv()` to read the highest index; if the highest index sent does not match the highest index received, P1 prints an error message and exits.

2.2 FIFOs

```
P1: mknod() → open() → read() & write() → close()
P2: open() → read() & write() → close()
```

P1 creates the FIFO and opens it twice throughout its execution - first to write the batch of 5 strings, and for the second time to read the highest index sent back by P2 (using read() and write() calls). P2 calls open() on the already created FIFO, and calls read() and write() to read the batch of 5 strings and then write back the highest index received. If the highest index sent does not match the highest index received, P1 prints an error message and exits. Both the programs then call close() and unlink() in order to destroy the created FIFO.

2.3 Shared Memory

```
P1: shmget() → shmat() → memcpy() & memset() → shmdt() → shmctl()
P2: shmget() → shmat() → memcpy() & memset() → shmdt()
```

P1 creates the shared memory segment; both P1 and P2 attach to the specific shared memory segment. P1 copies the set of 5 strings onto the shared memory segment by calling `memcpy()`; P2 reads the set of strings and then sends back the highest index received by copying the bytes onto the memory segment. If the highest index sent does not match the highest index received, P1 prints an error message and exits. Once all 50 strings have been sent, both program call `shmdt()` to detach from the shared memory segment, and P1 calls `shmctl()` to destroy the shared memory segment.

Times Taken

FIFO: 10.412593000 seconds
SHM: 10.378465000 seconds
UNIX SOCKETS: 10.424436000 seconds