Jatin Kumar Sharma
Roll no: 2020563
Operating System

Answer-2:

Makefile: All the part have their individual makefile that can be run using the command `make` and to clean the directory command `make clean` can be run. After doing make to run the the program ./1ans and ./2ans should be executed for sender (Server) and receiver(client) respectively.

Generating Random String: There is a single method being used to generate the random string of length 10, in all of the programs having definition:

```
void genRandomString(int n, char *res)
```

Here n: length of the required string and res is the pointer to the memory location where the generated random string should be stored. To generate 50 random strings, a char pointer array of length 50 is defined in the sender of each kind of implantation, and a for loop has been run to get the string from the genrandom function and stored in a local char array of length name “input” in each of the corresponding programs then is copied to the location a[i] of the initially defined char pointer array which been first allocated the required amount of memory using malloc.

Container Struct: All of the programs has a data structure defined named as container having definition as follows:

```
struct container
{
   /* data */
   int id;
   char data[10];
};
```

The struct is used to store the corresponding index from the array of strings of the length of 50 in variable id and the corresponding string in the data variable of the struct that is to be sent in the IPC process.

a). IPC using UNIX domain Sockets:

Server(socket1): In the main method of the program the array of random strings is being created using the method explained above and stored in the char array pointer “a”, after that an endpoint for communication has been created using the syscall. A local variable i been declared to track which message has been sent, an array of the data structure container of length 5 named ‘store’ is defined to store the chunks of 5 strings belonging from the index i to i+5 along with their indexes. This array containing the 5 strings is sent to the client using the ‘send’ syscall. This sent an array of the container data structure read by the client and the client returns the index of the last most index that have been received by it and send it back to the server(Procedure is explained in client part), and the value of i is increased to that value so that next time the strings to b dispatched could be from after this value the whole procedure executed in a while loop to

make sure the process continues and while loop ran till the last index not received from the client that is i become equal to 49.

Client(socket2): The client part of the socket has been implemented here using the proper syscalls as taught in the class, viz: socket creation then connected to the server. A variable log id defined which keeps the track of the index of the messages arrived till now, Now a while loop is implemented under the condition log < 49, the array of the length of 5 of the data structure container is declared which is passed to the read syscall and the data received by the client been stored in this array, now the data is printed using a for loop and the log is updated parallelly to the index of the latest received string with the condition that received indexes of the strings should be continuous otherwise the for loop will break and log would have the value of the last continuous index received and it will be sent back to the server using the send syscall.

After the whole process finishes the acquired socket in both of the programs will close.

b). IPC using FIFO: FIFOs are one-way communication channels hence to implement ipc of the said type two fifo have been created here one in FIFO1 and another in FIFO2.

Sender(FIFO1): In the main method of the program the array of random strings is being created using the method explained above and stored in the char array pointer "a", In this file the fifo1 is created and opened in write mode that will be used to send the strings and another fifo, fifo2 is created in the receiver(i.e FIFO2), which is opened in this file with rad access that will be used to read the highest index string received by the receiver. For sending purpose the method is similar to that of used in UNIX Socket. A local variable i been declared to track which message has been sent, an array of the data structure container of length 5 named 'store' is defined to store the chunks of 5 strings belonging from the index i to i+5 along with their indexes. This array containing the 5 strings is sent to the client using the 'send' syscall. This sent an array of the container data structure read by the client and the client returns the index of the last most index that have been received by it and send it back to the sender(Procedure is explained in receiver part) the setback index is read using fifo2 opened in sender in read mode, and the value of i is increased to that value so that next time the strings to b dispatched could be from after this value the whole procedure executed in a while loop to make sure the process continues and while loop ran till the last index not received from the client that is i become equal to 49.

Reciever(FIFO2): In this file the fifo2 is created and opened in write mode that will be used to recieve the strings and another fifo, fifo1 is created in the sennder(i.e FIFO1), which is opened in this file with read access that will be used to recieve string in the array of container of length of 5. For receiving purpos a variable log id defined which keeps the track of the index of the messages arrived till now, Now a while loop is implemented undr the condition log < 49, the array of the length of 5 of the data structure container is declared which is passed to the read syscall and the data received by the client been stored in this array, now the data is printed using

a for loop and the log is updated parallelly to the index of the latest received string with the condition that received indexes of the strings should be continuous otherwise the for loop will break and log would have the value of the last continuous index received and it will be sent back to the server using the send syscall.

After the whole process finishes the opened fifo in both of the programs had been closed.

c). IPC using Messege Queue:

Messege queue are one-way communication channels hence to implement ipc of the said type two queue should be created here one in **msgqueue1** and another in **msgqueue2**. Also to acquire a key for msg queue there should be a preexisting file path we have hence in makefile of the message queue there two file have been created named as idqueue and msgqueue to acquire key for two message queue one for sending and receiving the strings and one for sending and receiving the ids.

struct idbuffer: this data structure is created in response to send and receive the highest index received yet. The struct has an element mtype of the long data type(a default data type for msg to be passed by msgqueue) and a char array of length 1.

struct msgbuffer: this data structure is created in response to sending and receiving the packet of strings. The struct has an element mtype of the long datatype(a default data type for msg to be passed by msgqueue) and an array of container datatype of length 5. The question is done with the help of systemV message queues. We created two macros namely

Sender(msgqueue1): In the main method of the program the array of random strings is being created using the method explained above and stored in the char array pointer "a", In this file, first acquiring the key and then that key is used to create the msg queue and opened that will be used to send the strings and another msgqueue2 receiver(i.e msgqueue2) index string received by the receiver. For sending purpose the method is similar to that of used in UNIX Socket and FIFO. A local variable i been declared to track which message has been sent, an array of the data structure container of length 5 named 'store' is defined to store the chunks of 5 strings belonging from the index i to i+5 along with their indexes. This array containing the 5 strings is sent to the client using the msgsnd syscall. This sent an array of the container data structure read by the client and the client returns the index of the last most index that have been received by it and send it back to the sender(Procedure is explained in receiver part) the setback index is read using fifo2 opened in sender in reading mode, and the value of i is increased to that value so that next time the strings to b dispatched could be from after this value the whole procedure executed in a while loop to make sure the process continues and while loop ran till the last index not received from the client that is i become equal to 49.

Reciever(msgqueue2): In this file the msgqueue2 is created and opened in write mode that will be used to recieve the strings and another msgqueue, msgqueue1 is created in the sender(i.e msgqueue1), which is also acquired in this file will be used to receive string in the array of the container of the length of 5. For receiving purpose a variable log id defined which keeps the track of the index of the messages arrived till now, Now a while loop is implemented undr the condition log < 49, the array of the length of 5 of the data structure container is declared which is passed to the read syscall and the data received by the client been stored in this array, now the data is printed using a for loop and the log is updated parallelly to the index of the latest received string with the condition that received indexes of the strings should be continuous otherwise the for loop will break and log would have the value of the last continuous index received and it will be sent back to the sender using the msgsnd syscall.

After the whole process finishes the opened msgqueue in both of the programs had been closed.