Case	Description	Precondition	Steps	Exp Result	Actual Res (Pass/Fail)
TC1	Start program (Server)	(1) On Linux Terminal	(1) Go to folder with the application via terminal (2) type ./server -t server	Server mode started, program listening on message queue	Pass
TC2	Start program (Client)	(1) TC1 complete (2) On Linux Terminal	(1) Go to folder with the application via terminal (2) type ./server -t client -f "sometext4096.txt" -p 1024	(1) Server receives request from IPC (2) Client Application starts receiving data from server IPC	Pass
TC3	Multiple clients connecting at the same time	(1) Server running (2) TC2 successful (3) On multiple linux terminals	Client 1: ./server -t client -f "sometext4096.txt" -p 512 Client 2: ./server -t client -f "sometext4096.txt" -p 1024 Client 3: ./server -t client -f "sometext4096.txt" -p 2048	Server request obtained from all 3 clients, all 3 clients will start reading from message queue after successful init message from server	Pass
TC4	Multiple client requests in progress	TC3 started		All clients, with different priorities, will transfer at different speeds	Pass
TC5	Client request finished	TC4 finished		All clients will receive end message from server via message queue, and terminate accordingly. Result from each client with respect to number of message received and bytes received will be printed	Pass
TC6	Server request(s) complete	TC3 started		Server will print out after the file requested has finished being read, and will send last message to msg queue	pass
TC7	Incorrect usage of args (server)	(1) On Linux Terminal	(1) Go to folder with the application via terminal (2) type ./server -t server -f "sometext4096.txt"	Usage dialog will be printed to console, program ends	Pass
TC8	Incorrect usage of args (client)	(1) On Linux Terminal	(1) Go to folder with the application via terminal (2) type ./server -t client -f "sometext4096.txt"	Usage dialog will be printed to console, program ends	Pass

```
open queue ok, qid: 0
server Mode
server function running 16195
```

Figure 1-TC1 - Starting application, server shows the created/accessed msg queue ID and it's own PID on terminal

```
jacky@RIN   ~/Work/4985_4981/4981_Assn2   master
./server -t client -f "sometext4096.txt" -p 1024
open queue ok, qid: 0
client Mode
client thread initiated, passed in qid: 0
string to be sent to 0, length: 16
Client has sent: pid:26459
```

```
Init msg got size: 4112
Incoming: prior:1024, type:33268, pid:26459, incLen:16
msg:sometext4096.txt
srv transfer proc 26461 called for client proc: 26459
file open success
Transfer Requested: prior:1024, type:33268, pid:26459, incLen:16
msg:sometext4096.txt
Transfer packet size will be: 4
```

Figure 2- TC2: A single client connected to server (left – client view; right – server view)

```
Init msg got size: 4112
Incoming: prior:2048, type:33268, pid:26456, incLen:16
msg:sometext4096.txt
srv transfer proc 26458 called for client proc: 26456
file open success
Transfer Requested: prior:2048, type:33268, pid:26456, incLen:16
msg:sometext4096.txt
Transfer packet size will be: 2
Init msg got size: 4112
Incoming: prior:1024, type:33268, pid:26459, incLen:16
msg:sometext4096.txt
srv transfer proc 26461 called for client proc: 26459
file open success
Transfer Requested: prior:1024, type:33268, pid:26459, incLen:16
msg:sometext4096.txt
Transfer packet size will be: 4
Init msg got size: 4112
Incoming: prior:512, type:33268, pid:26462, incLen:16
msg:sometext4096.txt
srv transfer proc 26464 called for client proc: 26462
file open success
Transfer Requested: prior:512, type:33268, pid:26462, incLen:16
msg:sometext4096.txt
Transfer packet size will be: 8
```

Figure 3- TC3: Multiple Clients request to server (server view)

```
jacky@RIN  ~/Work/4985_4981/4981_Assn2  ⎇ master
./server -t client -f "sometext4096.txt" -p 512
open queue ok, qid: 0
client Mode
string to be sent to 0, length: 16
client thread initiated, passed in qid: 0
Client has sent: pid:26462
```

```
jacky@RIN  ~/Work/4985_4981/4981_Assn2  ⎇ master
./server -t client -f "sometext4096.txt" -p 1024
open queue ok, qid: 0
client Mode
client thread initiated, passed in qid: 0
string to be sent to 0, length: 16
Client has sent: pid:26459
```

```
jacky@RIN  ~/Work/4985_4981/4981_Assn2  ⎇ master
./server -t client -f "sometext4096.txt" -p 2048
open queue ok, qid: 0
client Mode
string to be sent to 0, length: 16
client thread initiated, passed in qid: 0
Client has sent: pid:26456
```

Figure 4- TC3: Multiple Clients requests to server (client view)

```
inc buffer filled: 213
inc buffer filled: 214
inc buffer filled: 215
```

```
inc buffer filled: 111
inc buffer filled: 112
inc buffer filled: 113
```

```
inc buffer filled: 46
inc buffer filled: 47
inc buffer filled: 48
```

Figure 5- TC4: Different speed for different priorities (left to right: priority = 512, 1024, 2048, respectively)

```
Srv end msg, totalbrecv: 1048320 totalmsg: 149761
```

```
Srv end msg, totalbrecv: 1048320 totalmsg: 349441
```

```
Srv end msg, totalbrecv: 1048320 totalmsg: 1048321
```

Figure 6- TC5: Different priority, different transfer results (left to right: priority = 512, 1024, 2048, respectively)

```
read file terminated
sending over last msg:
proc function finished
server proc 26464 finished
read file terminated
sending over last msg:
proc function finished
server proc 26461 finished
read file terminated
sending over last msg:
proc function finished
server proc 26458 finished
```

Figure 7- TC6: Server printing out end message,and sending end message to Clients (note in this figure, the server proc ID is the same as figure 3)

```
jacky@RIN  ~/Work/4985_4981/4981_Assn2  ⎇ master ●   ./server -t server -f "sometext4096.txt"
open queue ok, qid: 0
Run with options: -t server OR -t client -f filename -p int priority
```

Figure 8- TC7: Running server without correct params

```
jacky@RIN  ~/Work/4985_4981/4981_Assn2  ⎇ master ●   ./server -t clien
t -f "sometext4096.txt"
open queue ok, qid: 0
Run with options: -t server OR -t client -f filename -p int_priority
```

Figure 9 - TC8: Running Client without correct params