tw: total written
mTCR: maximum total can be read

Order of execution of code in Ensc351Part3-test.cpp and some output and details of the solution

which thread is blocked on cv Drain

threadT41 (priority 50)	threadT32 (priority 70 to 40 to 80 to 40)	threadT42 (priority 60)	daSktPr[0] - 3			daSktPr[1] - 4		
			tW/mTCR	pair	cvDrn	tW/mTCR	pair	cvDrn/cvR
			-	-	-	-	-	-
	mySocketpair(AF_LOCAL, SOCK_STREAM, 0, daSktPr);		0/0	4		0/0	3	/
	mySocketpair(AF_LOCAL, SOCK_STREAM, 0, daSktPr1);							
	mySocketpair(AF_LOCAL, SOCK_STREAM, 0, daSktPr2);							
	PE_NOT(myWrite(daSktPr[0], "abcd", 4), 4);		0/0	4		4/0	3	/
	posixThread threadT42(60, threadT42Func);							
	PE(myTcdrain(daSktPr[0])); // blocked		0/0	4		4/0	3	T32/
		PE_NOT(myWrite(daSktPr[1], "ijkl", 5), 5);	5/0	4		4/0	3	T32/
		posixThread threadT41(50, threadT41Func);						
		/* X */ myTcdrain(daSktPr[1]);//blocked	5/0	4	T42	4/0	3	T32/
/* A */ myReadcond(daSktPr[1], Ba, 20, 12, 0, 0); // blocked			5	4	T42	4/20	3	/T41
	setSchedPrio(40);							
	PE_NOT(myWrite(daSktPr[0], "123456789", 10), 10);		5/0	4	T42	14/20	3	/
/* finished: statement A: result was 14 Ba: abcd123456789 */			5/0	4	T42	0/0	3	/
myReadcond(daSktPr[1], Ba, 20, 0, 0, 0); // returned 0;			5/0	4	T42	0/0	3	/
myWrite(daSktPr[1], "Will not be read", 17);			22/0	4	T42	0/0	3	/
/* B */ myReadcond(daSktPr[1], Ba, 20, 12, 0, 0); // blocked			22/0	4	T42	0/20	3	/T41
	setSchedPrio(80);							
	PE_NOT (myWrite(daSktPr[0], "xyz", 4), 4);		22/0	4	T42	4/20	3	/T41
	PE(myTcdrain(daSktPr[0]));							
	PE(myClose(daSktPr[0])); // returned 0		22/0	-1		4/20	-2	/
	setSchedPrio(40);							
		/* finished: statement X: result was 0*/	22/0	-1		4/20	-2	/
		myTcdrain(daSktPr[1]);						
		threadT41.join();						
/* finished: statement B: result was 4 Ba: xyz */								
myReadcond(daSktPr[1], Ba, 20, 1, 0, 0)); // errno 104 “Reset”[i]								
myWrite(daSktPr1[1], "Will not be read", 17)								
/* C */ myReadcond(daSktPr1[1], Ba, 20, 1, 0, 0); // bl’ked								
	PE(myClose(daSktPr1[0]));							
/* finished: C: res was -1 errno 104: Connection reset by peer */								
myReadcond(daSktPr1[1], Ba, 20, 1, 0, 0)); // will return 0								
myWrite(daSktPr[1], "Added", 6); // ret -1 errno 32: Broken pipe								
/* D */ myRead(daSktPr2[1], Ba, 20); // blocked								
	PE_NOT(myWrite(daSktPr2[0], "mno", 4), 4);							
/* finished: statement D: result was 4 Ba: mno */								
/* E */ myRead(daSktPr2[1], Ba, 20);								
	PE(myClose(daSktPr2[0]));							
/* finished: statement E: result was 0*/								
myClose(daSktPr2[1]); // returned 0								
myClose(daSktPr1[1]); // returned 0								
myClose(daSktPr[1]); // returned 0								
myClose(daSktPr[1]); // ret -1 errno 9[ii]: Bad file descriptor								
myRead(daSktPr[1], Ba, 20); // ret -1 errno 9								
// …								
/* end of threadT41 */								
		/* end of threadT42 */						
	threadT42.join();							

[i] For this course, we will accept a return value of 0 in addition to errno 104 (Connection Reset by Peer)

[ii] The OS supplies the error string “Bad file descriptor” like this even though there is no “file” associated with the descriptor. I would have named it simply “Bad descriptor”, but files came first in the OS design.