Test example:

"serial versions test"

1 9  
U 0 w(x,3) w(y,7) w(z,10) c0;  
R 1 a0=r(x) c1;  
U 2 b2=r(z) w(x,b2) c2;  
R 3 c0=r(z) c1=r(z) c3;  
U 4 w(x,11) d2=r(y) d3=r(x) c4;  
R 5 e0=r(y) e1=r(x) e2=r(z) e3=r(z) e4=r(x) c5;  
U 6 w(x,13) w(z,14) c6;  
U 7 w(x,15) w(y,15) c7;  
R 8 b1=r(x) b2=r(z) c8;  
  
  
# versions ----  
# in the end of U0 : {("x",[("3",1)]),("y",[("7",1)]),("z",[("10",1)])}  
# in the end of R1 : {("x",[("3",1)]),("y",[("7",1)]),("z",[("10",1)])}  
## the version of x was updated by U2 and no one needs the older version because all are finished before U2  
# in the end of U2 : {("x",[("10",6)]),("y",[("7",1)]),("z",[("10",1)])}  
# in the end of R3 : {("x",[("10",6)]),("y",[("7",1)]),("z",[("10",1)])}  
## the version of x was updated by U4 and no one needs the older version because all are finished before U4  
# in the end of U4 : {("x",[("11",7)]),("y",[("7",1)]),("z",[("10",1)])}  
# in the end of R5 : {("x",[("11",7)]),("y",[("7",1)]),("z",[("10",1)])}  
## the version of x,z was updated by U6 and no one needs the older version because all are finished before U6  
# in the end of U6 : {("x",[("13",8)]),("y",[("7",1)]),("z",[("14",8)])}  
# in the end of U7 : {("x",[("15",9)]),("y",[("15",9)]),("z",[("14",8)])}  
# in the end of R8 : {("x",[("15",9)]),("y",[("15",9)]),("z",[("14",8)])}

This test check a serial run of the program, the above shows the test and an explanation of the final state of each transaction when they are being executed in a serial order.

The output of each step via the terminal:

\*\*\*\*\*fill in what the output looks like when done debugging\*\*\*\*\*

"Dead lock test" :

2 6  
U 0 w(x,3) w(y,6) w(z,8) c0;  
U 1 a0=r(x) w(y,18) a1=r(z) w(y,a1) a2=r(z) a3=r(y) c1;  
U 2 b0=r(y) w(x,5) b1=r(z) w(x,b1) b2=r(z) b3=r(y) w(y, b2) c2;  
U 3 c0=r(z) c1=r(z) w(z,7) c2=r(x) w(y,c2) w(z,c2) c3=r(z) c3;  
R 4 d0=r(x) d1=r(z) d2=r(y) d3=r(x) c4;  
R 5 e0=r(y) e1=r(x) e2=r(z) e3=r(z) e4=r(x) c5;  
  
  
# each line is a round of the round robin :  
#  
# U1:1 U2:1 U3:1 R4:1 R5:1  
# versions after round : {("x",[("3",1)]),("y",[("6",1)]),("z",[("8",1)])}  
# local var values {a0=3,b0=6,c0=8,d0=3,e0=6}  
  
# U1:2WAIT U2:RESET U3:2 R4:2 R5:2  
# versions after round : {("x",[("3",1)]),("y",[("6",1)]),("z",[("8",1)])}  
# local var values {a0=3,c0=8,d0=3,e0=6,c1=8,d1=8,e1=3}  
  
# U1:2 U2:1WAIT U3:3 R4:3 R5:3  
# versions after round : {("x",[("3",1)]),("y",[("6",1),("18",'uncommitted']),("z",[("8",1),("7",'uncommitted'])}  
# local var values {a0=3,c0=8,d0=3,e0=6,c1=8,d1=8,e1=3,d2=6,e2=8}  
  
# U1:3 U2:1WAIT U3:4 R4:4 R5:4  
# versions after round : {("x",[("3",1)]),("y",[("6",1),("18",'uncommitted']),("z",[("8",1),("7",'uncommitted'])}  
# local var values {a0=3,c0=8,d0=3,e0=6,c1=8,d1=8,e1=3,d2=6,e2=8,a1=7,c2=3,d3=3,e3=8}  
  
# U1:4 U2:1WAIT U3:5WAIT R4:5 R5:5  
# versions after round : {("x",[("3",1)]),("y",[("6",1),("18",'uncommitted'),("7",'uncommitted)]),("z",[("8",1),("7",'uncommitted'])}  
# local var values {a0=3,c0=8,e0=6,c1=8,e1=3,e2=8,a1=7,c2=3,e3=8,e4=3}  
  
# U1:5RSET U2:1 U3:5WAIT R5:6  
# versions after round : {("x",[("3",1)]),("y",[("6",1)]),("z",[("8",1),("7",'uncommitted'])}  
# local var values {c0=8,c1=8,c2=3,b0=6}  
  
# U1:1 U2:2RSET U3:5  
# versions after round : {("x",[("3",1)]),("y",[("6",1),("3",'uncommitted']),("z",[("8",1),("7",'uncommitted'])}  
# local var values {c0=8,c1=8,c2=3,a0=3}  
  
# U1:2WAIT U2:1WAIT U3:6  
# the same transaction updated the local uncommitted update for var z - only the last one was left  
# versions after round : {("x",[("3",1)]),("y",[("6",1),("3",'uncommitted']),("z",[("8",1),("3",'uncommitted'])}  
# local var values {c0=8,c1=8,c2=3,a0=3}  
  
# U1:2WAIT U2:1WAIT U3:7  
# versions after round : {("x",[("3",1)]),("y",[("6",1),("3",'uncommitted']),("z",[("8",1),("3",'uncommitted'])}  
# local var values {c0=8,c1=8,c2=3,a0=3,c3=3}  
  
# U1:2WAIT U2:1WAIT U3:8  
# U3 committed - it got ts 4 (after 0 , and 2 readers), no reader uses the previous values - only latest value is left  
# versions after round : {("x",[("3",1)]),("y",[("3",4]),("z",[("3",4])}  
# local var values {a0=3}  
  
# U1:2 U2:1WAIT  
# versions after round : {("x",[("3",1)]),("y",[("3",4),("18",'uncommitted']),("z",[("3",4])}  
# local var values {a0=3}  
  
# U1:3 U2:1WAIT  
# versions after round : {("x",[("3",1)]),("y",[("3",4),("18",'uncommitted']),("z",[("3",4])}  
# local var values {a0=3,a1=3}  
  
# U1:4 U2:1WAIT  
# the same transaction updated its own local var ! (18->3)  
# versions after round : {("x",[("3",1)]),("y",[("3",4),("4",'uncommitted']),("z",[("3",4])}  
# local var values {a0=3,a1=3}  
  
# U1:5 U2:1WAIT  
# versions after round : {("x",[("3",1)]),("y",[("3",4),("4",'uncommitted']),("z",[("3",4])}  
# local var values {a0=3,a1=3,a2=3}  
  
# U1:6 U2:1WAIT  
# versions after round : {("x",[("3",1)]),("y",[("3",4),("4",'uncommitted']),("z",[("3",4])}  
# local var values {a0=3,a1=3,a2=3,a3=4}  
  
# U1:7 U2:1WAIT  
# U1 committed and there are no other readers in the system - keep latest value  
# versions after round : {("x",[("3",1)]),("y",[("4",5]),("z",[("3",4])}  
# local var values {}  
  
# U2:1  
# versions after round : {("x",[("3",1)]),("y",[("4",5]),("z",[("3",4])}  
# local var values {b0=5}  
  
# U2:2  
# versions after round : {("x",[("3",1),("3",uncommitted)]),("y",[("4",5]),("z",[("3",4])}  
# local var values {b0=5}  
  
# U2:3  
# versions after round : {("x",[("3",1),("3",uncommitted)]),("y",[("4",5]),("z",[("3",4])}  
# local var values {b0=5,b1=3}  
  
# U2:4  
# the same transaction updated its own local var ! (5->3)  
# versions after round : {("x",[("3",1),("3",uncommitted)]),("y",[("4",5]),("z",[("3",4])}  
# local var values {b0=5,b1=3}  
  
# U2:5  
# versions after round : {("x",[("3",1),("3",uncommitted)]),("y",[("4",5]),("z",[("3",4])}  
# local var values {b0=5,b1=3,a2=3}  
  
# U2:6  
# versions after round : {("x",[("3",1),("3",uncommitted)]),("y",[("4",5]),("z",[("3",4])}  
# local var values {b0=5,b1=3,a2=3}  
  
# U2:7  
# versions after round : {("x",[("3",1),("3",uncommitted)]),("y",[("4",5]),("z",[("3",4])}  
# local var values {b0=5,b1=3,a2=3,b3=4}  
  
# U2:8  
# versions after round : {("x",[("3",6)]),("y",[("4",5]),("z",[("3",4])}  
# local var values {}

This test checks a round robin run of the program, the above shows the test and an explanation of the final state of each round.  
in this test there are aborts due to deadlocks and waits.

The output of each step via the terminal:

\*\*\*\*\*fill in what the output looks like when done debugging\*\*\*\*\*