Appendix A: Interpreter test programs

```
./2dautomata.occ
SEQ
    [2][2][2]INT rule:
    rule [0][0][0] := 0
    rule [0][0][1] := 1
    rule [0][1][0] := 1
    rule [0][1][1] := 1
    rule [1][0][0] := 1
    rule [1][0][1] := 0
    rule [1][1][0] := 0
    rule [1][1][1] := 0
    [32]INT yCoord:
    [32]INT me:
    SEQ i = [0 FOR 31]
        me[i] := 0
    me[16] := 1
    [32]INT myLeft:
    [32]INT myRight:
    [32]CHAN OF INT left:
    [32]CHAN OF INT right:
    PAR
        SEQ
            yCoord[0] := 0
            WHILE yCoord[0] < 32
                SEQ
                    GRAPHICS[0][yCoord[0]] ! me[0]+7
                    left[0] ! me[0]
                    left[31] ? myLeft[0]
                    right[31] ! me[0]
                    right[0] ? myRight[0]
                    me[0] := rule[myLeft[0]][me[0]][myRight[0]]
                    yCoord[0] := yCoord[0] + 1
        PAR i = [1 FOR 31]
            SEQ
                yCoord[i] := 0
                WHILE yCoord[i] < 32
                    SEQ
                        GRAPHICS[i][yCoord[i]] ! me[i]+7
                        left[i-1] ? myLeft[i]
                        left[i] ! me[i]
                        right[i] ? myRight[i]
                        right[i-1] ! me[i]
                        me[i] := rule[myLeft[i]][me[i]][myRight[i]]
                        yCoord[i] := yCoord[i] + 1
./while.occ
---
xs = [5,0]
```

```
---
SEQ
    INT x:
    x := 0
    SEQ
        WHILE x < 5
            x := x + 1
        SERIAL ! x
        WHILE x >= 1
            x := x - 1
        SERIAL ! x

./seqreplicator2.occ
---
xs = [1,2,3,4,5,6,7,8,9,10]
---
SEQ
    CHAN OF INT chan:
    INT y:
    PAR
        WHILE TRUE
            SEQ i = [1 FOR 10]
                chan ! i
        WHILE TRUE
            SEQ
                chan ? y
                SERIAL ! y
./ring.occ
SEQ
    [32]INT me:
    SEQ i = [0 FOR 31]
        me[i] := 0
    me[16] := 1
    [32]INT myLeft:
    [32]CHAN OF INT left:
    PAR
        SEQ
            left[0] ! me[0]
            left[31] ? myLeft[0]
        PAR i = [1 FOR 31]
            SEQ
                left[i-1] ? myLeft[i]
                left[i] ! me[i]
./assignment.occ
---
xs = [1,2,3,4,5]
---
SEQ
    INT x:
```

```
    x := 1
    INT y:
    SEQ
        y := 5
        SERIAL ! x
        x := 2
        SERIAL ! x
        x := 3
        SERIAL ! x
        x := 4
        SERIAL ! x
        x := 1
        SERIAL ! y
./alt_replicator.occ
---
sort xs = [1,2,3,4,5]
---
SEQ
    CHAN OF INT individuals[5]:
    PAR
        WHILE TRUE
            ALT i = [1 FOR 5]
                individuals[i] ? x
                    SERIAL ! x
        PAR i = [1 FOR 5]
            individuals[i] ! i
./slowprime.occ
---
filter (\x -> x == 0) = 0
---

SEQ
    INT x:
    x := 37
    CHAN chan:
    PAR i = [2 FOR 37]
        SEQ
            INT c:
            c := x
            WHILE c >= 0
                c := c - i
            SERIAL ! c
./arrays_noreplicator.occ
SEQ
    [2][3]INT slots:
    [3]CHAN OF INT chans:
    SEQ
        PAR
            chans[0] ! 0
```

```
            chans[1] ! 1
            chans[2] ! 2
            chans[0] ? slots[1][0]
            chans[1] ? slots[0][1]
            chans[2] ? slots[0][2]
        SEQ
            SERIAL ! slots[1][0]
            SERIAL ! slots[0][1]
            SERIAL ! slots[0][2]./ifmustbebool.occ
---
error
---
SEQ
    INT x:
    x := 1
    IF x
        INT y:
./precedence.occ
---
xs == [1,2]
---
SEQ
    IF 4+5*2>10
        SERIAL ! 1
    IF 10<4+5*2
        SERIAL ! 2
./if.occ
---
xs = [1,2,3,4,5]
---
SEQ
    INT x:
    x := 1
    SEQ
        IF
            (1 >= 0) AND (1 >= 1)
                SEQ
                    IF
                        0 >= 1
                            SERIAL ! 0
                    SERIAL ! 1
        IF
            (0 <= 0) AND (0 <= 1)
                SEQ
                    IF
                        1 <= 0
                            SERIAL ! 0
                    SERIAL ! 2
        IF
```

```
            1 > 0
                SEQ
                    IF
                        0 > 1
                            SERIAL ! 0
                    SERIAL ! 3
        IF
            0 < 1
                SEQ
                    IF
                        1 < 0
                            SERIAL ! 0
                    SERIAL ! 4
        IF
            0 = 0
                SEQ
                    IF
                        0 = 1
                            SERIAL ! 0
                    SERIAL ! 5
```

```
./countdownup.occ
---
filter xs (\x -> x > 0) == sort (filter xs (\x -> x > 0))
filter xs (\x -> x < 0) == reverse (sort (filter xs (\x -> x < 0)))
---
SEQ
    CHAN OF INT left:
    CHAN OF INT right:
    INT x:
    INT y:
    INT z:
    x := 1
    y := 1
    PAR
        WHILE TRUE
            SEQ
                right ! x
                x := (x+1)
        WHILE TRUE
            SEQ
                left ! y
                y := (y-1)
        WHILE TRUE
            ALT
                TRUE & left ? z
                    SERIAL ! z
                right ? z
                    SERIAL ! z
```

```
./no_arrays_ring.occ
SEQ
    INT x:
    INT y:
    INT z:
    x := 0
    y := 1
    z := 2
    INT xl:
    INT yl:
    INT zl:
    CHAN OF INT xleft:
    CHAN OF INT yleft:
    CHAN OF INT zleft:
    PAR
        SEQ
            yleft ! x
            xleft ? xl
        SEQ
            yleft ? yl
            zleft ! y
        SEQ
            zleft ? zl
            xleft ! z

./unassigned2.occ
---
error
---
SEQ
    INT x:
    WHILE x < 5
        x := x + 1
./unassigned.occ
---
error
---
SEQ
    INT x:
    x := x + 1
./par_replicator.occ
---
sort xs = [1,2,3,4,5]
---
PAR i = [1 FOR 5]
    SERIAL ! i
./seqreplicator.occ
---
xs = [2,3,4,5]
```

```
---
SEQ i = [2 FOR 5]
    SERIAL ! i
./nested_par.occ
PAR
    PAR i = [0 FOR 31]
        SERIAL ! i
    PAR i = [0 FOR 31]
        SERIAL ! (100+i)
./undeclared.occ
---
error
---
SEQ
    INT y:
    y := 1
    x := y
./sameamount.occ
---
length (filter (\x -> x == 1) xs) == 5
length (filter (\x -> x == 0) xs) == 5
---

SEQ
    INT x:
    CHAN chan:
    PAR
        SEQ i = [0 FOR 4]
            chan ! 1
        SEQ i = [0 FOR 4]
            chan ! 1
        WHILE TRUE
            SEQ
                chan ? x
                SERIAL ! x
./alternate.occ
---
xs = [0,1,0,1..]
---
SEQ
    INT x:
    CHAN chan:
    PAR
        WHILE TRUE
            SEQ
                chan ! 0
                chan ! 1
        WHILE TRUE
            SEQ
```

```
                chan ? x
                SERIAL ! x
./whilemustbebool.occ
---
error
---
SEQ
    INT x:
    x := 1
    WHILE x
        INT y:
./io_block.occ
---
xs = [2]
---
SEQ
    INT x:
    x := 1
    CHAN OF INT chan:
    CHAN OF INT chan2:
    PAR
        SEQ
            chan ! x
            SERIAL ! 0
        SEQ
            chan2 ? x
            SERIAL ! 1
        SERIAL ! 2

./alt.occ
---
filter (\x -> x != 3) (drop ((find 3 xs)-1) xs) == []
length xs == find 3 xs
---
SEQ
    [3]CHAN OF INT myChan:
    INT seenthree:
    seenthree := 0
    INT x:
    PAR
        WHILE TRUE
            ALT
                seenthree < 1 & myChan[0] ? x
                    SKIP
                seenthree < 1 & myChan[1] ? x
                    SKIP
                myChan[2] ? x
                    seenthree := 1
        PAR
```

```
            WHILE TRUE
                myChan[0] ! 1
            WHILE TRUE
                myChan[1] ! 2
            WHILE TRUE
                myChan[2] ! 3
./arrays.occ
---
xs = [0,1,2]
---
SEQ
    [3]INT slots:
    [3]CHAN OF INT chans:
    SEQ
        PAR
            PAR i = [0 FOR 2]
                chans[i] ! i
            PAR i = [0 FOR 2]
                chans[i] ? slots[i]
        SEQ
            SERIAL ! slots[0]
            SERIAL ! slots[1]
            SERIAL ! slots[2]
```