117 lab6

part 1 :

One)

1:

S1 T1 S2 T2 S3 S3.1 -- T2 does not unify

becauseT1: B already returned true, for the second one T2: B cannot be used again to represent different variables.

2.

S1 S2 S3 S3.1 T2 T1 -- Does not unify

because S1 may match T2, S2 may match T1. both cannot match to the original “T”

3.

S2 T2 S1 T1 S3 S3.1 -- T2 does not unify

because T2 and T1 both represent the same “B”, but different values, once S2 matched T2, S1 and T1 cannot match.cause S2 and T2 return false, but s3 and s3.1 only If B is true, then skip browse B. that’s why cannot unify

4.

S2 T1 S3 S3.1 S1 T2 -- display true, but can’t unify B in value creation

because S2 matched to true, then went S3 and S3.1. After that, we recalled thread (S1), but it cannot unify the first thread.

5.

S1 T2 S3 S3.1 T2 T1 -- Does not unify

B already represents false, then EXU1 (B is true) then skip, but B is not true, so cannot skip browse B, which means they are not unify.

6.

S1 T2 S3 S3.1 S2 T1 can’t uniify B in value creation

from thread S1 then T2 (B)= false, after that, pass through S3 and S3.1, only if B is true then skip browse B. So cannot unify B in value creation.

7. S1 S2 T1 T2 S3 S3.1 cannot unify

because S1 match to T1 , then S2 matches T2, but they are not the same variable. it will cause error

8. S2 S1 T2 T1 S3 S3.1 does not unify

It's the same reason with the No.7

9 S1 T2 S2 T1 S3 S3.1 does not unify

also same reason like the previous 2, S1 S2 and T1 T2 all match before S3 and s3.1. In this case, True and False all matched, so they cannot unify

Two)

When we run with Infinity, we got Y T2 T1 are unbound, because Infinity is not an exact variable, and cannot get an exact value. when we use Finite 1, we get Y:3, T2:3 and T1:Unbound. This means we only run for the first thread for each one

Three)

```
local Z in
  Z = 3
  thread local X in
    X = 1
    skip Browse X
    skip Browse X
    skip Basic
    skip Browse X
    skip Browse X
    skip Basic
    skip Browse X
    end
  end
  thread local Y in
    Y = 2
    skip Browse Y
    skip Basic
    skip Browse Y
    skip Browse Y
    skip Browse Y
    skip Basic
    skip Browse Y
    end
  end
  skip Browse Z
  skip Browse Z
  skip Browse Z
  skip Basic
  skip Browse Z
  skip Browse Z
  skip Basic
 end
```

output:

```
*Hoz> runFullT (Finite 4) "declarative threaded" "thread.txt" "thread.out"
X : 1

X : 1

Y : 2

Z : 3
```

Z : 3

Z : 3

X : 1

X : 1

Y : 2

Y : 2

Y : 2

Z : 3

Z : 3

X : 1

Y : 2

Four)
quantum 5 is the minimum quantum number that suspension to occur

Five)
fib1_sugar.txt fib2_sugar.txt fib1_thread.txt

X	Times			X	times		X	times
13	(7.04 secs, 2,024,370, 704 bytes)			1000	(20.61 secs, 6,352,390,928 bytes)		9	(18.30 secs, 186,235,616 bytes)
14	(16.37 secs, 5,200,605, 584 bytes)			1100	(21.20 secs, 7,671,166,176 bytes)		10	(30.33 secs, 438,212,280 bytes)
15	(42.42 secs, 13,450,004 ,016 bytes)			1200	(28.02 secs, 9,114,292,728		11	(52.11 secs, 1,064,753,880 bytes)

					bytes)			
16	(117.49 secs, 34,936,173 ,128 bytes)			1300	(35.06 secs, 10,681, 707,712 bytes)		12	(94.76 secs, 2,646,688,8 88 bytes)
				1400	(36.36 secs, 12,373, 442,944 bytes)			
				1500	(43.02 secs, 14,189, 496,728 bytes)			
				1600	(50.94 secs, 16,129, 871,712 bytes)			
				1700	(56.39 secs, 18,194, 566,696 bytes)			
				1750	(57.19 secs, 19,273, 533,728 bytes)			
				1760	(62.76 secs, 19,493, 090,128 bytes)			
				1755	(60.66 secs, 19,383, 172,576 bytes)			

				1754	(59.34 secs, 19,361, 194,104 bytes)			

a)input X is a number that functions needs to work out, the bigger input, the more times used. During the solving times, it has a lot of bytes. bigger bytes will spends more time on it

b) fib0 = 1, fib1= 1, fib2=2 , fib3=3 ,fib4=5, fib5=8, fib6=13, fib7=21
fib(n)= fib(n-1)+fib(n-3)

Part2

```
local N L P F Result Producer OddFilter Consumer in
thread
Producer = proc {$ N Limit Out}
if (N<Limit) then T N1 in
Out = (N|T)
N1 = (N + 1)
{Producer N1 Limit T}
else Out = nil
end
end
end
thread
OddFilter = proc {$ P Out}
case P of nil then
Out = nil
[]'|'(1:X 2:Y) then T in
if ((X mod 2) == 0) then
Out = (X|T)
{OddFilter Y T}
else
{OddFilter Y Out}
end
end
end
end
thread
Consumer = fun {$ P} in
case P of nil then 0
[]'|'(1:X 2:Y) then R in
(X+{Consumer Y})
end
end
```

```
end
N = 0
L = 100
{Producer N L P}
{OddFilter P F}
skip Browse F
Result = {Consumer F}
skip Browse Result
end
```