1a)

0: not c goes back to itself c goes to 1 double circle

1: not a goes back to 0 a goes to 2

2: not t goes back to 0 t goes to 3

3: have an ‘any’ go to 3 double circle

bi)

I had states A-J

ii)

I had 6 states

iii) no clue

ci)

3a)

transExp :: Exp -> [Register] -> [Instr]

transExp (Plus e1 e2) (r1:r2:rs) =

transExp e1 (r1:rs) ++

transExp e2 (r2:rs) ++

[Add (Reg r2) (Reg r1)]

transExp (Num n) (r1 : rs) =

[Mov (ImmNum n) (Reg r1)]

transExp (Ref (Var id)) (r1:rs) =

[Mov (Abs id) (Reg r1)]

transExp (Ref (Array id e)) (r1:r2:rs) =

transExp e (r1:rs) ++

[

Mul (ImmNum 4) (Reg r1)

Mov (ImmName id) (Reg r2)

Add (Reg r1) (Reg r2)

Mov (Ind r2) (Reg r1)

]

b)

transStat :: Stat -> [Register] -> [Instr]

transStat (Assign (Var id) e) (r1:rs) =

transExp e (r1:rs) ++

[Mov (Reg r1) (ImmName id)]

transStat (Assign (Array id arrIdxExp) rhsExp) (r1: r2 : rs) =

transExp rhsExp (r1: r2 : rs) ++ # RHS expression => R1

transExp arrIdxExp (r2 : rs) ++ # Array index => R2

[

Mul (ImmNum 4) (Reg r2), # Offset = 4 \* arrIdx => R2

Add (ImmName id) (Reg r2), # indexed addr = base addr + offset => R2

Mov (Reg r1) (Ind r2) # Ind R2 => Reg R1

]