七、按照下述给出的翻译模式，写出

while (a<b) do

if A or B then x:=y+z else x:=y-z

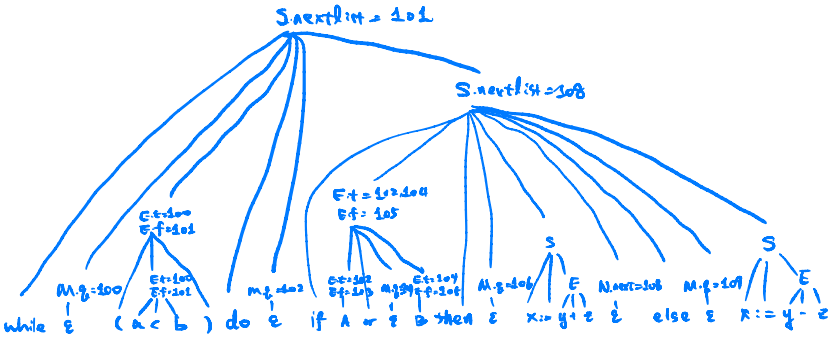
的四元式序列，约定四元式序列的起始标号为 100。（12 分）

产生式	语义规则
E→E1 or M E2	{ backpatch(E1.falselist, M.quad); E.truelist:=merge(E1.truelist, E2.truelist); E.falselist:=E2.falselist }
E→E1 and M E2	{ backpatch(E1.truelist, M.quad); E.truelist:=E2.truelist; E.falselist:=merge(E1.falselist,E2.falselist) }
E→(E1)	{ E.truelist:=E1.truelist; E.falselist:=E1. falselist}
E→id1 relop id2	{ E.truelist:=makelist(nextquad); E.falselist:=makelist(nextquad+1); emit(‘j’ relop.op ‘,’ id 1.place ‘,’ id 2.place ‘,’ ‘0’); emit(‘j, －, －, 0’) }
E→id	{ E.truelist:=makelist(nextquad); E.falselist:=makelist(nextquad+1); emit(‘jnz’ ‘,’ id .place ‘,’ ‘－’ ‘,’ ‘0’); emit(‘ j, -, -, 0’) }
S→if E then M1 S1 N else M2 S2	{backpatch(E.truelist, M1.quad); backpatch(E.falselist, M2.quad); S.nextlist:=merge(S1.nextlist, N.nextlist, S2.nextlist) }
S→while M1 E do M2 S1	{backpatch(S_1.nextlist, M_1.quad); backpatch(E.truelist, M_2.quad); S.nextlist:=E.falselist emit(‘j,－,－,’ M_1.quad) }
S→id:=E	{ emit(‘:=’ ‘,’ E.place ‘,’ ‘－’ ‘,’id .place)}
E→E1 op E2	{ E.place:=newtemp; emit(op ‘,’ E1.place ‘,’ E2.place ‘,’ E.place)}
M→ε	{ M.quad:=nextquad }
N→ε	{ N.nextlist:=makelist(nextquad); emit(‘j,－,－,0’) }

产生式	语义规则
E→E1 or M E2	{ backpatch(E1.falselist, M.quad); E.truelist:=merge(E1.truelist, E2.truelist); E.falselist:=E2.falselist }
E→E1 and M E2	{ backpatch(E1.truelist, M.quad); E.truelist:=E2.truelist; E.falselist:=merge(E1.falselist,E2.falselist) }
E→(E1)	{ E.truelist:=E1.truelist; E.falselist:=E1. falselist}
E→id1 relop id2	{ E.truelist:=makelist(nextquad); E.falselist:=makelist(nextquad+1); emit('j' relop.op ',' id 1.place ',' id 2.place ',' '0'); emit('j, −, −, 0') }
E→id	{ E.truelist:=makelist(nextquad); E.falselist:=makelist(nextquad+1); emit('jnz' ',' id .place ',' '−' ',' '0'); emit(' j, -, -, 0') }
S→if E then M1 S1 N else M2 S2	{backpatch(E.truelist, M1.quad); backpatch(E.falselist, M2.quad); S.nextlist:=merge(S1.nextlist, N.nextlist, S2.nextlist) }
S→while M1 E do M2 S1	{backpatch(S_1.nextlist, M_1.quad); backpatch(E.truelist, M_2.quad); S.nextlist:=E.falselist emit('j,−,−,' M_1.quad) }
S→id:=E	{ emit(':=' ',' E.place ',' '−' ',' id .place)}
E→E1 op E2	{ E.place:=newtemp; emit(op ',' E1.place ',' E2.place ',' E.place)}
M→ε	{ M.quad:=nextquad }
N→ε	{ N.nextlist:=makelist(nextquad); emit('j,−,−,0') }

while (a<b) do

if A or B then x:=y+z else x:=y-z



100 j<, a, b, 102
101 j, −, −, 0
102 jnz, A, −, 106
103 j, −, −, 104
104 jnz, B, −, 106
105 j, −, −, 109
106 +, y, z, T1
107 :=, T1, −, x
108 j, −, −, 100
109 −, y, z, T2
110 :=, T2, −, x
111 j, −, −, 100
112

八、考虑文法　E→T+T | T×T

T→a | b

（1）列出该文法拓广文法的所有 LR(0)项目；（2 分）

（2）构造该文法的 LR(0)项目集规范族及识别活前缀的 DFA；（4 分）

（3）构造该文法的 LR(0)分析表；（4 分）

状态	action					GOTO	
	+	×	a	b	#	E	T
0							

（4）判断该文法是否是 LR(0)文法，并说明理由；（2 分）

(1) 拓广文法：

$E' \to E$

$E \to T+T \mid T \times T$

$T \to a \mid b$

LR(0)项目：

$E' \to \cdot E$, $E' \to E \cdot$

$E \to \cdot T+T$, $E \to T \cdot +T$, $E \to T+\cdot T$, $E \to T+T \cdot$

$E \to \cdot T \times T$, $E \to T \cdot \times T$, $E \to T \times \cdot T$, $E \to T \times T \cdot$

$T \to \cdot a$, $T \to a \cdot$

$T \to \cdot b$, $T \to b \cdot$

(2) 项目集族：

I_0: $E' \to \cdot E$, $E \to \cdot T+T$, $E \to \cdot T \times T$, $T \to \cdot a$, $T \to \cdot b$

I_1: $E' \to E \cdot$

I_2: $E \to T \cdot +T$, $E \to T \cdot \times T$

I_3: $E \to T+\cdot T$, $T \to \cdot a$, $T \to \cdot b$

I_4: $T \to a \cdot$

I_5: $T \to b \cdot$

I_6: $E \to T \times \cdot T$, $T \to \cdot a$, $T \to \cdot b$

I_7: $E \to T+T \cdot$

I_8: $E \to T \times T \cdot$

(3)

State	+	×	a	b	#	GOTO E	GOTO T
0			S4	S5		1	2
1					acc		
2	S3	S6					
3			S4	S5			7
4	r3	r3	r3	r3	r3		
5	r4	r4	r4	r4	r4		
6			S4	S5			8
7	r1	r1	r1	r1	r1		
8	r2	r2	r2	r2	r2		

comment from 2/8/2024 : 考试真心不难，好多人做完还剩 20min ~ 1h.