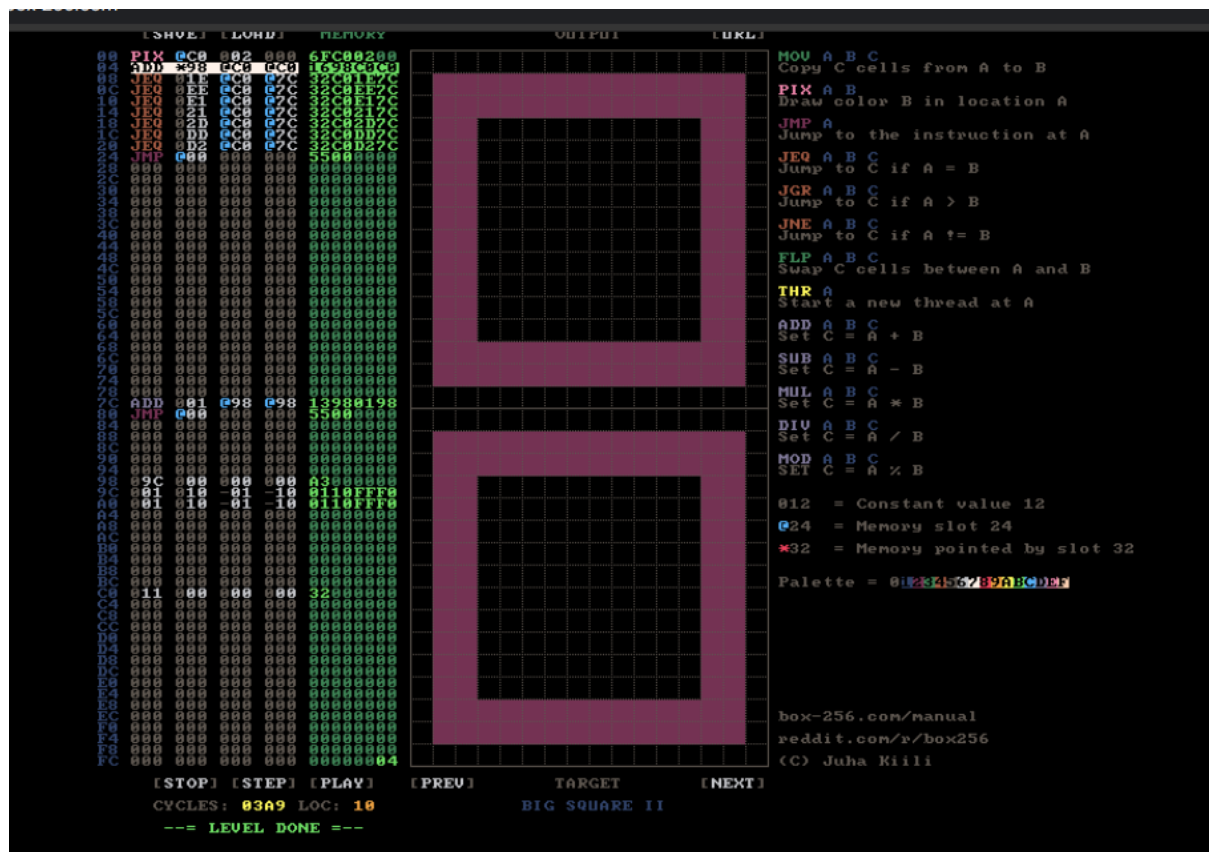


Código[illegible]

```

000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
011 000 000 000
000 000 000 000
000 000 000 000

```



• Ejercicio 3

Código

```

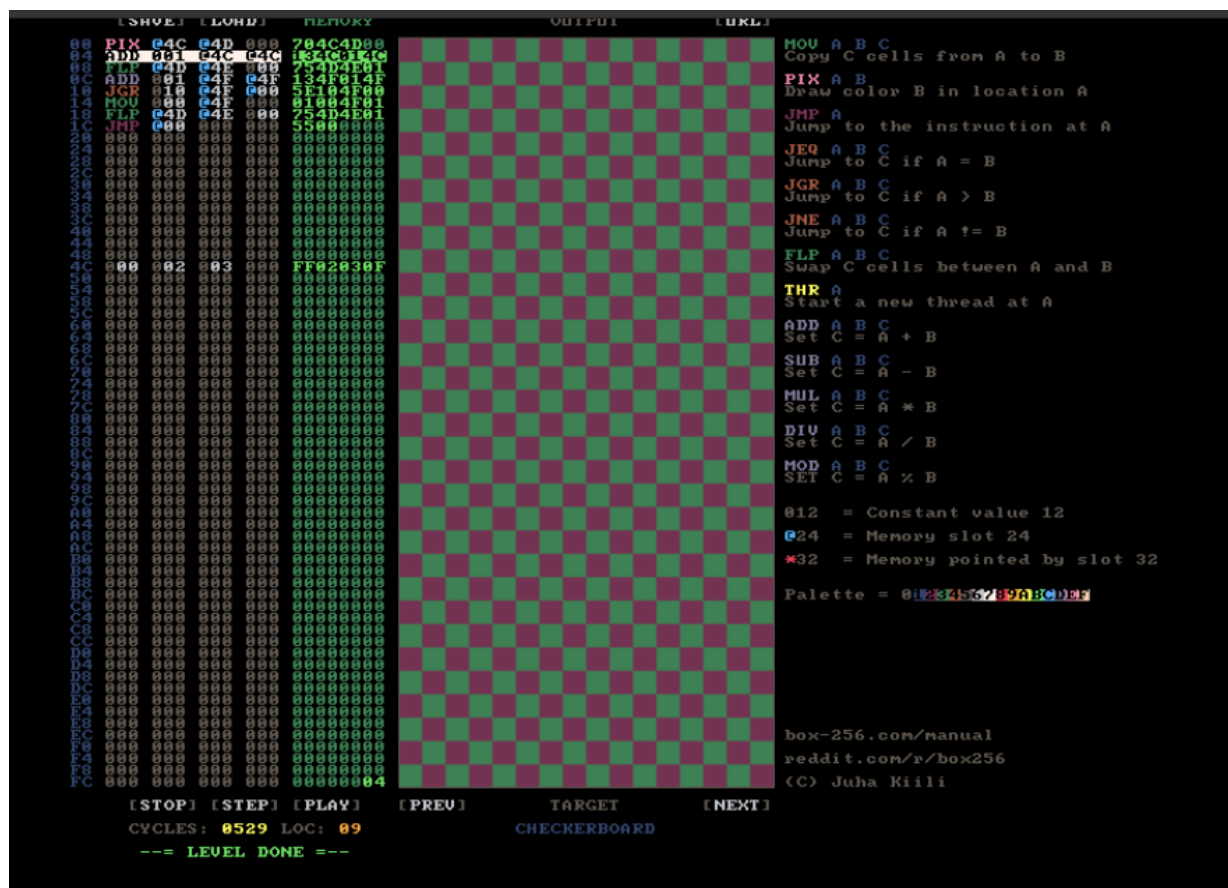
PIX @4C @4D 000
ADD 001 @4C @4C
FLP @4D @4E 000
ADD 001 @4F @4F
JGR 010 @4F @00
MOV 000 @4F 000
FLP @4D @4E 000
JMP @00 000 000
000 000 000 000
000 000 000 000

```

```

000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 002 003 000
000 000 000 000
000 000 000 000

```



• Ejercicio 4

Código

```

PIX @4C @4D 000
ADD 001 @4C @4C
ADD 001 @4F @4F
JGR 002 @4F @00
FLP @4D @4E 000
MOV 000 @4F 000
ADD 001 @50 @50

```

[STOP]	[LOAD]	MEMORY	OUTPUT	[LURL]
00	PIX 04C 04D 000	704C4D00		MOV A B C
04	ADD 001 04C 04C	104C0010		Copy C cells from A to B
08	ADD 001 04F 04F	134F014F		PIX A B
12	JGR 002 04F 000	5E024F00		Draw color B in location A
16	FLP 04D 04E 000	754D4E01		
20	MOV 000 04F 000	01004F01		JMP A
24	ADD 001 050 050	13500150		Jump to the instruction at A
28	JGR 010 050 000	5E105000		JEQ A B C
32	FLP 04D 04E 000	754D4E01		Jump to C if A = B
36	MOV 000 050 000	01005001		JGR A B C
40	JMP 000 000 000	55000000		Jump to C if A > B
44	000 000 000 000	00000000		JNE A B C
48	000 000 000 000	00000000		Jump to C if A != B
52	000 000 000 000	00000000		FLP A B C
56	000 000 000 000	00000000		Swap C cells between A and B
60	000 000 000 000	00000000		THR A
64	000 000 000 000	00000000		Start a new thread at A
68	000 000 000 000	00000000		ADD A B C
72	000 000 000 000	00000000		Set C = A + B
76	000 000 000 000	00000000		SUB A B C
80	000 000 000 000	00000000		Set C = A - B
84	000 000 000 000	00000000		MUL A B C
88	000 000 000 000	00000000		Set C = A * B
92	000 000 000 000	00000000		DIV A B C
96	000 000 000 000	00000000		Set C = A / B
100	000 000 000 000	00000000		MOD A B C
104	000 000 000 000	00000000		SET C = A % B
108	000 000 000 000	00000000		012 = Constant value 12
112	000 000 000 000	00000000		024 = Memory slot 24
116	000 000 000 000	00000000		*32 = Memory pointed by slot 32
120	000 000 000 000	00000000		Palette = 0123456789ABCDEF
124	000 000 000 000	00000000		
128	000 000 000 000	00000000		
132	000 000 000 000	00000000		
136	000 000 000 000	00000000		
140	000 000 000 000	00000000		
144	000 000 000 000	00000000		
148	000 000 000 000	00000000		
152	000 000 000 000	00000000		
156	000 000 000 000	00000000		
160	000 000 000 000	00000000		
164	000 000 000 000	00000000		
168	000 000 000 000	00000000		
172	000 000 000 000	00000000		
176	000 000 000 000	00000000		
180	000 000 000 000	00000000		
184	000 000 000 000	00000000		
188	000 000 000 000	00000000		
192	000 000 000 000	00000000		
196	000 000 000 000	00000000		
200	000 000 000 000	00000000		
204	000 000 000 000	00000000		
208	000 000 000 000	00000000		
212	000 000 000 000	00000000		
216	000 000 000 000	00000000		
220	000 000 000 000	00000000		
224	000 000 000 000	00000000		
228	000 000 000 000	00000000		
232	000 000 000 000	00000000		
236	000 000 000 000	00000000		
240	000 000 000 000	00000000		
244	000 000 000 000	00000000		
248	000 000 000 000	00000000		
252	000 000 000 000	00000000		
256	000 000 000 000	00000000		
260	000 000 000 000	00000000		
264	000 000 000 000	00000000		
268	000 000 000 000	00000000		
272	000 000 000 000	00000000		
276	000 000 000 000	00000000		
280	000			

- **Ejercicio 5**

Código

```
PIX @8C @8D 000
ADD 001 @8C @8C
ADD 001 @8E @8E
JGR 004 @8E @00
MOV 000 @8E 000
ADD 00C @8C @8C
ADD 001 @8F @8F
JGR 004 @8F @00
ADD 004 @8C @8C
MOV 000 @8F 000
ADD 001 @8D @8D
ADD @90 001 @90
JGR 004 @90 @00
MOV 000 @90 000
MOV 011 @8C 000
PIX @8C 000 000
ADD 001 @8C @8C
PIX @8C 000 000
ADD 00F @8C @8C
ADD 001 @8E @8E
JGR 002 @8E @3C
MOV 000 @8E 000
ADD 024 @8C @8C
JMP @3C 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 008 000 000
000 000 000 000
000 000 000 000
```

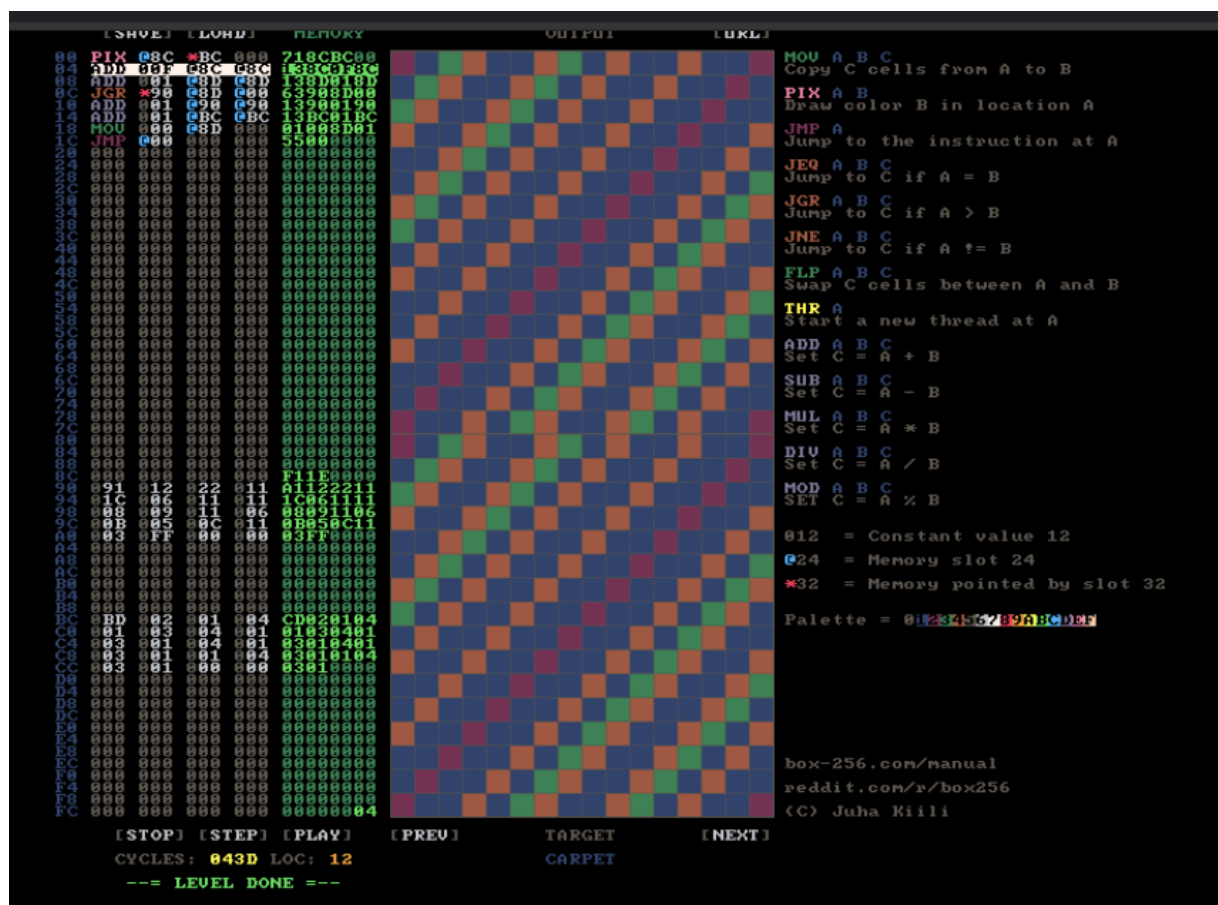

[illegible]

Código[illegible]


```

000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
0BD 002 001 004
001 003 004 001
003 001 004 001
003 001 001 004
003 001 000 000
000 000 000 000
000 000 000 000

```



• Ejercicio 8

Código

```

PIX @8C @34 000
ADD 001 @8C @8C
PIX @8C @35 000
ADD 001 @8C @8C
PIX @8C @35 000
ADD 001 @8C @8C
PIX @8C @35 000

```

[illegible]

```
PIX @8C 008 000
ADD *F0 @8C @8C
ADD 001 @8D @8D
JGR 003 @8D @00
MOV 000 @8D 000
ADD 001 @F0 @F0
ADD 001 @8E @8E
JGR 003 @8E @00
ADD *F4 @8C @8C
MOV 000 @8E 000
```

[illegible]

```

000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
0F1 011 -01 -10
0F5 040 004 000
000 000 000 000
000 000 000 000

```

The screenshot shows the Box256 emulator interface. The top section displays registers (A, B, C) and memory slots (000-00F). The middle section shows a list of instructions being executed, such as `MOV A B C`, `PIX A B`, `JMP A`, etc. The bottom section shows the current state of the program, including the number of cycles (01E6) and the location (LOC: 18). The Sierpinski triangle is visualized as a red pattern on a black background.

- Ejercicio 10

Código

```

PIX @F0 001 000
ADD 010 @F0 @F0
ADD 001 @F1 @F1
JGR 010 @F1 @00
MOV 000 @F1 000
ADD @0D -01 @0D
MOV 000 @F0 000
ADD 011 @F0 @F0
ADD 011 @19 @19

```

[illegible]

```

000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
0E9 081 041 0C1
0C9 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000

```



• Ejercicio 11

Código

```

PIX @F0 00A 000
ADD 001 @F0 @F0
JGR 0FF @F0 @00
MOV *B8 @F0 000
PIX @F0 000 000
ADD 001 @F0 @F0
ADD 001 @F1 @F1

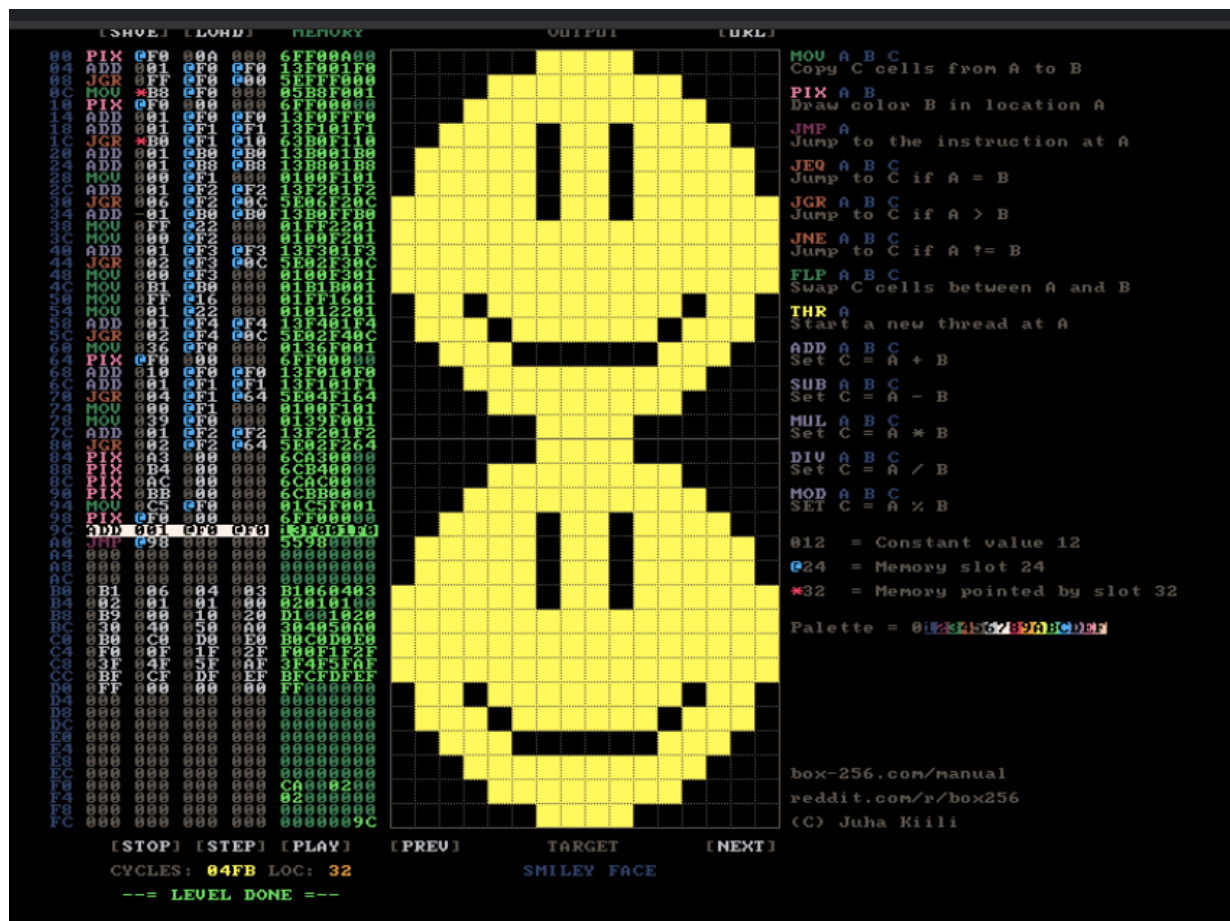
```

JGR *B0 @F1 @10
ADD 001 @B0 @B0
ADD 001 @B8 @B8
MOV 000 @F1 000
ADD 001 @F2 @F2
JGR 006 @F2 @0C
ADD -01 @B0 @B0
MOV 0FF @22 000
MOV 000 @F2 000
ADD 001 @F3 @F3
JGR 002 @F3 @0C
MOV 000 @F3 000
MOV 0B1 @B0 000
MOV 0FF @16 000
MOV 001 @22 000
ADD 001 @F4 @F4
JGR 002 @F4 @0C
MOV 036 @F0 000
PIX @F0 000 000
ADD 010 @F0 @F0
ADD 001 @F1 @F1
JGR 004 @F1 @64
MOV 000 @F1 000
MOV 039 @F0 000
ADD 001 @F2 @F2
JGR 002 @F2 @64
PIX 0A3 000 000
PIX 0B4 000 000
PIX 0AC 000 000
PIX 0BB 000 000
MOV 0C5 @F0 000
PIX @F0 000 000
ADD 001 @F0 @F0
JMP @98 000 000
000 000 000 000
000 000 000 000
000 000 000 000
0B1 006 004 003
002 001 001 000
0B9 000 010 020
030 040 050 0A0
0B0 0C0 0D0 0E0
0F0 00F 01F 02F
03F 04F 05F 0AF


```

0BF 0CF 0DF 0EF
OFF 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000

```



• Ejercicio 12

Código

```

MOV *DD @F8 000
PIX @F8 *AE 000
ADD 001 @F8 @F8
ADD 001 @F9 @F9
JGR *C2 @F9 @04

```

MOV 000 @F9 000
ADD 001 @C2 @C2
ADD 001 @DD @DD
ADD 001 @FA @FA
JGR *B8 @FA @00
MOV 000 @FA 000
ADD 001 @B8 @B8
ADD 001 @AE @AE
ADD 001 @FB @FB
JGR 009 @FB @00
PIX 035 004 000
PIX 033 004 000
PIX 043 004 000
PIX 054 004 000
PIX 04A 004 000
PIX 029 004 000
PIX 039 004 000
PIX 0A6 00A 000
PIX 0A9 00A 000
PIX 0A4 008 000
PIX 0AB 008 000
PIX 076 00C 000
PIX 086 00C 000
PIX 079 00C 000
PIX 089 00C 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 000 000
000 000 0AF 008
00F 008 00F 00C
004 000 00C 004
0B9 002 005 003
003 001 002 003
004 003 0C3 005
009 007 00A 00B
009 006 008 00A
00C 00C 00C 00C
008 00A 00C 002

	[SAVE]	[LOAD]	MEMORY	OUTPUT	[LUKE]
00	MOV	*DD	0F8 000 05DDF801		MOV A B C
04	PIX	0F8	*AE 000 71F8AE00		Copy C cells from A to B
08	ADD	001	0F8 000 13F801F8		PIX A B
12	JGR	*C2	0F9 004 63C2F904		Draw color B in location A
16	MOV	000	0F9 000 0100F901		JMP A
20	ADD	001	0C2 000 13C201C2		Jump to the instruction at A
24	ADD	001	0AD 000 0100AD01		JEQ A B C
28	JGR	*B8	0FA 000 63B8FA00		Jump to C if A = B
32	MOV	000	0FA 000 0100FA01		JGR A B C
36	ADD	001	0B8 000 13B801B8		Jump to C if A > B
40	JGR	000	0FB 000 5E5FB000		JNE A B C
44	PIX	033	004 000 6C330400		Jump to C if A != B
48	PIX	043	004 000 6C430400		FLP A B C
52	PIX	054	004 000 6C540500		Swap C cells between A and B
56	PIX	0A0	004 000 6CA00A00		THR A
60	PIX	029	004 000 6C290200		Start a new thread at A
64	PIX	039	004 000 6C390300		ADD A B C
68	PIX	0A6	00A 000 6CA60A00		Set C = A + B
72	PIX	0A9	00A 000 6CA90A00		SUB A B C
76	PIX	0A4	008 000 6CA40800		Set C = A - B
80	PIX	0AB	008 000 6CAB0800		MUL A B C
84	PIX	076	00C 000 6C760C00		Set C = A * B
88	PIX	086	00C 000 6C860C00		DIV A B C
92	PIX	079	00C 000 6C790C00		Set C = A / B
96	PIX	0B9	00C 000 6CB90C00		MOD A B C
100	000	000	000 000 00000000		Set C = A % B
104	000	000	000 000 00000000		
108	000	000	000 000 00000000		
112	000	000	000 000 00000000		
116	000	000	000 000 00000000		
120	000	000	000 000 00000000		
124	000	000	000 000 00000000		
128	000	000	000 000 00000000		
132	000	000	000 000 00000000		
136	000	000	000 000 00000000		
140	000	000	000 000 00000000		
144	000	000	000 000 00000000		
148	000	000	000 000 00000000		
152	000	000	000 000 00000000		
156	000	000	000 000 00000000		
160	000	000	000 000 00000000		
164	000	000	000 000 00000000		
168	000	000	000 000 00000000		
172	000	000	000 000 00000000		
176	000	000	000 000 00000000		
180	000	000	000 000 00000000		
184	000	000	000 000 00000000		
188	000	000	000 000 00000000		
192	000	000	000 000 00000000		
196	000	000	000 000 00000000		
200	000	000	000 000 00000000		
204	000	000	000 000 00000000		
208	000	000	000 000 00000000		
212	000	000	000 000 00000000		
216	000	000	000 000 00000000		
220	000	000	000 000 00000000		
224	000	000	000 000 00000000		
228	000	000	000 000 00000000		
232	000	000	000 000 00000000		
236	000	000	000 000 0000000		