

HintsAssignVI

SEE:

Program: ADD.RAM

		MEMORY									
		00	10	20	30	40	50	60	70	80	90
FETCH Instr		0	+001	+030		+001					
IR<-MEM[11]; PC<-(PC)+1		1		+031							
ACCC		2		+230							
IR		3		+431							
PC		4		+332							
INPUT		5		+132							
OUTPUT		6		+910							
		7									
		8									
		9									+700

<ESC>=exit <SPACEBAR>=do-a-halfcycle speed=FAST

...IN TERMS OF WHERE YOU ARE IN THE PROGRAM:

Editing: ADD.RAM

- 1: +000 -
- 2: +000 - PUPPOSE: Add two integers.
- 3: +000 - INPUT: two integers.
- 4: +000 - OUTPUT: sum of the two integers.
- 5: +000 - -
- 6: +000 - -
- 7: +000 - -
- 8: +000 - -
- 9: +000 - -
- 0: +030 - (30)<-input | A=input (YOU'VE INPUT A '1')
- 1: +031 - (31)<-input | B=input <= **(YOU ARE NOW HERE)**
- 2: +230 - ACC<-(30)
- 3: +431 - ACC<-(ACC)+(31) | C=A+B
- 4: +332 - (32)<-(ACC)
- 5: +132 - output (32) | output C
- 6: +910 - PC<-10 and quit | quit
- 7: +000 - -
- 8: +000 - -
- 9: +000 - -
- 0: +000 - (See also MULT.RAM)
- 1: +000 -
- 2: +000 -
- 3: +000 -

3D) REPEATING THE SAME PROCEDURE AS IN 3B), 3C) WILL ENABLE YOU TO ENTER THE SECOND INTEGER TO ADD (I.E., THE NUMBER 2). YOU SHOULD THEN SEE:

Program: ADD.RAM			HintsAssignVI									
			00	10	20	30	MEMORY		60	70	80	90
FETCH Instr			0	+001	+030		+001					
IR<-MEM[12]; PC<-(PC)+1			1		+031		+002					
			2		+230							
ACC	IR	PC	3		+431							
+ 0 0 0 0	0 3 1	1 2	4		+332							
			5		+132							
			6		+910							
			7									
			8									
			9								+700	

<ESC>=exit <SPACEBAR>=do-a-halfcycle speed=FAST

...IN TERMS OF WHERE YOU ARE IN THE PROGRAM:

Editing: ADD.RAM

- 1: +000 -
- 2: +000 - PUPOSE: Add two integers.
- 3: +000 - INPUT: two integers.
- 4: +000 - OUTPUT: sum of the two integers.
- 5: +000 - -
- 6: +000 - -
- 7: +000 - -
- 8: +000 - -
- 9: +000 - -
- 0: +030 - (30)<-input | A=input (YOU'VE INPUT A '1')
- 1: +031 - (31)<-input | B=input (YOU'VE INPUT A '2')
- 2: +230 - ACC<-(30) | <= **(YOU ARE NOW HERE)**
- 3: +431 - ACC<-(ACC)+(31) | C=A+B
- 4: +332 - (32)<-(ACC)
- 5: +132 - output (32) | output C
- 6: +910 - PC<-10 and quit | quit
- 7: +000 - -
- 8: +000 - -
- 9: +000 - -
- 0: +000 - (See also MULT.RAM)
- 1: +000 -
- 2: +000 -
- 3: +000 -

4) REPEATEDLY HIT THE SPACEBAR UNTIL THE PROGRAM TERMINATES (WITH THE OUTPUT "3") YOU SHOULD SEE (WHICH CARRIES THE "<= (*YOU ARE HERE*)" POINTER DOWN EACH LINE) :

		HintsAssignVI									
		00	10	20	30	40	50	60	70	80	90
FETCH Instr		0	+001	+030		+001					
IR<-MEM[10]; PC<-(PC)+1		1		+031		+002					
		2		+230		+003					
ACC	IR	3		+431							
+ 0 0 0 3	9 1 0	4		+332							
		5		+132							
INPUT	OUTPUT	6		+910							
1. +001	1. +003	7									
2. +002	>	8									
>		9									+700

<ESC>=exit HALT!

...IN TERMS OF WHERE YOU ARE IN THE PROGRAM:

Editing: ADD.RAM

```

1: +000 -
2: +000 - PUPOSE: Add two integers.
3: +000 - INPUT: two integers.
4: +000 - OUTPUT: sum of the two integers.
5: +000 - -
6: +000 - -
7: +000 - -
8: +000 - -
9: +000 - -
0: +030 - (30)<-input      | A=input   (YOU'VE INPUT A '1')
1: +031 - (31)<-input      | B=input   (YOU'VE INPUT A '2')
2: +230 - ACC<-(30)
3: +431 - ACC<-(ACC)+(31) | C=A+B
4: +332 - (32)<-(ACC)
5: +132 - output (32)    | output C  (OUTPUT = '3')
6: +910 - PC<-10 and quit | quit      <= **(YOU ARE NOW HERE)**
7: +000 - -
8: +000 - -
9: +000 - -
0: +000 - (See also MULT.RAM)
1: +000 -
2: +000 -
3: +000 -

```

5) NOW TRY TO REPEAT STEPS 1)-4) WITH DIFFERENT INPUTS, AND DIFFERENT RUN OPTIONS (CONTINUOUS, ETC.)

II. To play with 'count-up':

- 1) SELECT COUNTUP.RAM FROM "SELECT PROGRAM"
- 2) SELECT RUN, YOU SHOULD SEE:

HintsAssignVI

Program: COUNTUP.RAM		MEMORY									
		00	10	20	30	40	50	60	70	80	90
FETCH Instr		0	+001	+200							
IR<-MEM[10]; PC<-(PC)+1		1		+309							
		2		+109							
ACC IR PC		3		+400							
+ 0 0 0 0		4		+711							
+ 0 0 0		5									
+ 1 0		6									
INPUT OUTPUT		7									
+ ...		8									
+ ...		9									+700

<ESC>=exit Help Tools Step-by-step Continuous EditMemory EditComt

IN TERMS OF THE PROGRAM:

Editing: COUNTUP.RAM

```

1: +000 -
2: +000 -      COUNT-UP Program:
3: +000 -      Outputs the positive integers,
4: +000 -      in order, beginning with 1.
5: +000 - -
6: +000 -      (Demonstrates the Unconditional Jump instruction, 7)
7: +000 - -
8: +000 -      PC starts at cell 10.
9: +000 - -
0: +200 -      Clear ACC & add contents of cell 00.      <= (*YOU ARE HERE*)
1: +309 -      Store contents of ACC in cell 09. <----|
2: +109 -      Output contents of cell 09.           | = loop
3: +400 -      Add contents of cell 00 to ACC.       |
4: +711 -      Jump to cell 11. -----|
5: +000 - -
6: +000 - -
7: +000 -      (For Conditional Jump instruction 8, see COUNTDN.RAM)
8: +000 -
9: +000 -
0: +000 -
1: +000 -
2: +000 -
3: +000 -
ESC=>exit Help EditKeys:<RETURN><SPACE><BACKSPACE> ↑ ↓ ← → <PGUP><PGDOWN>

```

3) AS INDICATED, THIS PROGRAM SIMPLY ADDS "1" RECURSIVELY TO THE INITIAL VALUE (1). FOR EXAMPLE, BREAKING THE LOOP (RUNNING IN "C" MODE) AT THE 425TH STEP:

Program: COUNTUP.RAM		MEMORY									
		00	10	20	30	40	50	60	70	80	1701
											90

```

                                HintsAssignVI
EXECUTE Instr                   0 | +001 | +200 |
+-----+-----+-----+-----+
MEM[09]<-(ACC)                  1 |      | +309 |
+-----+-----+-----+-----+
                                2 |      | +109 |
ACC      IR      PC            3 |      | +400 |
+-----+-----+-----+-----+
+|0|4|2|6|  |3|0|9|  |1|2|      4 |      | +711 |
+-----+-----+-----+-----+
                                5 |      |      |
INPUT      OUTPUT              6 |      |      |
+-----+-----+-----+-----+
>          >                    7 |      |      |
+-----+-----+-----+-----+
                                8 |      |      |
                                9 | +425 |      | +715 |
+-----+-----+-----+-----+

```

<ESC>=exit <SPACE-BAR>=resume

4) NOTE THE ROLE OF "7" COMMAND (LINE 14). THIS IS WHERE THE RECURSION (ADD 1 TO TEMPORARILY STORED VALUE OCCURS.

III. To play with 'count-down:'

- 1) SELECT COUNTDN.RAM FROM "SELECT PROGRAM"
- 2) SELECT RUN, YOU SHOULD SEE:

```

Program: COUNTDN.RAM
                                MEMORY
                                00  10  20  30  40  50  60  70  80  90
                                0
FETCH Instr                      0 | +001 | -005 |
+-----+-----+-----+-----+
IR<-MEM[11]; PC<-(PC)+1         1 |      | +210 |
+-----+-----+-----+-----+
                                2 |      | +400 |
ACC      IR      PC            3 |      | +309 |
+-----+-----+-----+-----+
+|0|0|0|0|  |0|0|0|  |1|1|      4 |      | +109 |
+-----+-----+-----+-----+
                                5 |      | +812 |
INPUT      OUTPUT              6 |      | +911 |
+-----+-----+-----+-----+
>          >                    7 |      |      |
+-----+-----+-----+-----+
                                8 |      |      |
                                9 |      |      | +700 |
+-----+-----+-----+-----+

```

ESC>=exit Help Tools Step-by-step Continuous EditMemory EditComt

HintsAssignVI

...IN TERM OF THE PROGRAM:

```

Editing: COUNTDN.RAM
1: +000 - COUNTDOWN Program
2: +000 - Counts down from negative integer in cell 10,
3: +000 - outputting each successive smaller integer until 0.
4: +000 - -
5: +000 - -
6: +000 - (Demonstrates Conditional Jump instruction 8)
7: +000 - -
8: +000 - PC starts at cell 11.
9: +000 - -
0: -005 - = countdown kernel (a negative integer)
1: +210 - kernel to ACC
2: +400 - add 01
3: +309 - store ACC in cell 09
4: +109 - output cell 09
5: +812 - test ACC: + -> go on to 16; - -> jump to 12 -- = branch point
6: +911 - halt & reset to cell 11
7: +000 - -
8: +000 - (For Unconditional Jump instruction 7, see COUNTUP.RAM)
9: +000 -
0: +000 -
1: +000 -
2: +000 -
3: +000 -
ESC=>exit Help EditKeys:<RETURN><SPACE><BACKSPACE> ↑ | ← → <PGUP><PGDOWN>
    
```

3) RUNNING THE PROGRAM (WHETHER STEP-BY-STEP OR CONTINUOUS) WILL REPEATEDLY ADD "1" TO WHAT WAS ORIGINALLY IN THE COUNTDOWN KERNEL (THE INTEGER -5, BY DEFAULT) IT SHOULD STOP WHEN OUTPUT = 0, WHICH SHOULD LOOK LIKE:

Program: COUNTDN.RAM			MEMORY									
			00	10	20	30	40	50	60	70	80	90
FETCH Instr			0	+001	-005							
IR<-MEM[11]; PC<-(PC)+1			1		+210							
			2		+400							
ACC	IR	PC	3		+309							
+ 0 0 0 0	9 1 1	1 1	4		+109							
			5		+812							
			6		+911							
			7									
			8									
			9	+000							+700	
INPUT		OUTPUT										
+.....+		+.....+										
		1. -004										
		2. -003										
		3. -002										
		4. -001										
		5. +000										
>		>										
+.....+		+.....+										

ESC=>exit HALT!

IN TERMS OF THE PROGRAM:

HintsAssignVI

```

1: +000 -   COUNTDOWN Program
2: +000 -   Counts down from negative integer in cell 10,
3: +000 -   outputting each successive smaller integer until 0.
4: +000 -   -
5: +000 -   -
6: +000 -   (Demonstrates Conditional Jump instruction 8)
7: +000 -   -
8: +000 -   PC starts at cell 11.
9: +000 -   -
10: -005 -   = countdown kernel (a negative integer)
11: +210 -   kernel to ACC
12: +400 -   add 01
13: +309 -   store ACC in cell 09
14: +109 -   output cell 09
15: +812 -   test ACC: + -> go on to 16; - -> jump to 12 -- = branch poi
16: +911 -   halt & reset to cell 11
17: +000 -   -
18: +000 -   (For Unconditional Jump instruction 7, see COUNTUP.RAM)
19: +000 -   -
20: +000 -   -
21: +000 -   -
22: +000 -   -
23: +000 -   -
<ESC>=exit Help EditKeys:<RETURN><SPACE><BACKSPACE>↑↓←→<PGUP><PGDOWN>

```

4) NOTE THE ROLE OF "8" COMMAND (LINE 16). IF OUTPUT < 0, IT RETURNS TO LINE 12, TO ADD "1". OTHERWISE, IT HALTS.

5) TRY RUNNING COUNTDOWN.RAM WITH DIFFERENT COUNTDOWN KERNELS (SELECT "m" TO EDIT MEMORY) FOR INSTANCE, FOR COUNTDOWN KERNEL = -100:

Program: COUNTDN.RAM			MEMORY									
			00	10	20	30	40	50	60	70	80	90
FETCH Instr			0	+001	-100							
IR<-MEM[11]; PC<-(PC)+1			1		+210							
			2		+400							
ACC	IR	PC	3		+309							
+ 0 0 0 0	9 1 1	1 1	4		+109							
			5		+812							
			6		+911							
			7									
			8									
			9								+700	
INPUT	OUTPUT											
.....											
.....	1. -004											
.....	2. -003											
.....	3. -002											
.....	4. -001											
.....	5. +000											
>	>											

<ESC>=exit Help Tools Step-by-step Continuous EditMemory EditComt

HintsAssignVI

SELECTING "C" FOR CONTINUOUS RUN, YOUR EVENTUAL OUTPUT WILL BE:

Program: COUNTDN.RAM			MEMORY									
			00	10	20	30	40	50	60	70	80	424 90
FETCH Instr			0	+001	-100							
IR<-MEM[11]; PC<-(PC)+1			1		+210							
			2		+400							
ACC IR PC			3		+309							
+ 0 0 0 0			4		+109							
+ 9 1 1			5		+812							
+ 1 1			6		+911							
INPUT OUTPUT			7									
+ ...			8									
100. -005			9	+000								+700
101. -004												
102. -003												
103. -002												
104. -001												
105. +000												
>												
+ ...												

<ESC>=exit HALT!

IV) THE MULT.RAM PROGRAM:

Editing: MULT.RAM

```

3: +000 -
4: +000 - -
5: +000 - -
6: +000 - -
7: +000 - -
8: +000 - -
9: +000 - -
0: +030 - (30)<-input | A=input
1: +604 - ACC<-(ACC)/10000
2: +331 - (31)<-(ACC) | Z=0
3: +032 - (32)<-input | B=input
4: +232 - ACC<-(32)
5: +500 - ACC<-(ACC)-10
6: +332 - (32)<-(ACC) | ---B
7: +822 - goto 22, if (ACC)<0 | If B<0, goto (22) | ---Add A to Z,
8: +231 - ACC<-(31) | | B times.
9: +430 - ACC<-(ACC)+(30)
0: +331 - (31)<-(ACC) | Z=Z+A
1: +714 - goto 14 | Goto (14)
2: +131 - output (31) | Output Z
3: +910 - PC<-10 and quit | Quit
4: +000 -
5: +000 -
ESC>=exit Help EditKeys:<RETURN><SPACE><BACKSPACE> ↑ ↓ ← → <PGUP><PGDOWN>

```

REMARK: SINCE MULTIPLICATION IS COMMUTATIVE (I.E. AB=BA) ADDING A TO Z B TIMES IS THE SAME AS ADDING B TO Z A TIMES. MODIFY THE ABOVE ROUTINE SO THAT

HintsAssignVI

FOR WHATEVER INPUTS (N,M), THEN: A=N, B=M WHENEVER N > M.

CONSIDER THE LOGIC OF EQUALS.RAM

```
1: +000 - -
2: +000 -   EQUALS Tester:
3: +000 -   Tests 2 positive integers for equality:
4: +000 -   A > B -> Print out A, then B
5: +000 -   B > A -> Print out B, then A
6: +000 -   A = B -> Print out 0
7: +000 - -
8: +000 -   PC starts at 10.
9: +000 - -
10: +030 -   read in A to 30
11: +031 -   read in B to 31
12: +230 - --- clear ACC & add A
13: +531 -   | subtract B from A
14: +820 -   | test ACC: + -> go on to 15; - -> jump to 20 (A < B)
15: +500 -   | subtract 1 from ACC
16: +823 - --- test ACC: + -> go on to 17 (B < A); - -> jump to 23 (A = B)
17: +130 - --|
18: +131 -   |-- A > B
19: +910 - --|
20: +131 - ---
21: +130 -   |-- A < B
22: +910 - ---
23: +132 -   -- A = B
<ESC>=exit  Help  EditKeys:<RETURN><SPACE><BACKSPACE> ↑ | ← → <PGUP><PGDOWN>
```

SIMPLY WORK IN ITS LOGIC TO MULT.RAM (CUT & PASTE THE APPROPRIATE LINES FROM EQUALS.RAM BETWEEN LINE 13 AND 14 IN MULT.RAM. MAKE SURE YOU ADJUST THE LINE NUMBERS ACCORDINGLY (IN THE BATCH FROM EQUALS.RAM, SO THAT IT REFERS TO THE APPROPRIATE LINES IN MULT.RAM)
