Middle East Technical University
Department of Computer Engineering

CENG 111

Fall ’97

Worksheet 1

REGULATIONS

Due date: 28 October, Tuesday *(Not subject to postpone)*

Submission: You will be given a separate sheet of questions on 21 October, Tuesday (during the lecture). Fill it out and submit it to any of the teaching assistents of the course *(Do not forget to fill in the name box!)*

Team: There is **no** teaming up. The homework has to be done and turned in individually.

Cheating: All parts involved (source(s) and receiver(s)) get zero.

PROBLEM

In this homework you will be experimenting with a given process on number sequences. This process resemble the action of a simplified Von Neumann Machine. You will observe that the number sequence (which will change from problem to problem) acts like a machine code program loaded into the memory of a Von Neumann machine. The two registers $\mathcal{R}_1$, $\mathcal{R}_2$ and the instruction pointer $\mathcal{I}$, the only three internals of the process, carry out the functions of the data, address, condition and the instruction registers of a Von Neumann machine.

Our process (lets name it from now on as the **HW-1 machine**) works on a sequence of integers each of which are in the range $[-127, +127]$. Here is an example for such a sequence:

1	14	2	33	16	8	8	0	0
0	1	2	3	4	5	6	7	8

If we are speaking about the value of the 3rd element in the sequence then we will denote this by enclosing the 3 into square brackets, like [3]. In the example above [3] is 33.

Furthermore, the HW-1 machine has three registers which we name as $\mathcal{R}_1$, $\mathcal{R}_2$ an $\mathcal{I}$. Each one of $\mathcal{R}_1$ and $\mathcal{R}_2$ can hold an integer in the range $[-127, +127]$. Storing negative values into $\mathcal{I}$ is not allowed, it can hold any integer in the range $[0, +255]$ (though in your exercises the values of $\mathcal{I}$ will be about 10-30, at most). The square bracket notation applies for these registers as well. Namely $[\mathcal{R}_1]$ means the content of the $\mathcal{R}_1$th element in the sequence. So, for instance, if $\mathcal{R}_1$ has at any moment the value 3 and if at that moment the 3rd sequence element is `33` (as it is in the example above) then $[\mathcal{R}_1]$ refers that `33`.

The interesting point is that the values in the sequence as well as the values in the registers can be changed freely to new values. When this is the case the former value is erased and the new value is substituted in that place. We will call this action an *assignment* and will represent it by the following notation:

$$\textit{place} \leftarrow \textit{new value}$$

Here are some assignment examples:

$\mathcal{R}_1 \leftarrow 5$	$\mathcal{R}_1$ is set to the value 5
$\mathcal{R}_1 \leftarrow \mathcal{R}_2$	$\mathcal{R}_1$ is set to the same value $\mathcal{R}_2$ is holding now. *(<u>Attention:</u> This does not mean that any following changes on $\mathcal{R}_2$ will effect the value stored in $\mathcal{R}_1$)*
$\mathcal{R}_2 \leftarrow [2]$	$\mathcal{R}_2$ is set to the 2nd value in the sequence.
$[0] \leftarrow \mathcal{R}_1$	The 0th (zeroth) value in the sequence is changed to be the same value that is in $\mathcal{R}_1$.
$\mathcal{I} \leftarrow \mathcal{I} + 1$	The content of $\mathcal{I}$ is incremented by one.

The number of changes is not limited and can be performed as many times as desired on any register or sequence element.

Given a sequence, HW-1 starts with the $\mathcal{I}$ register having 0 (zero) value and the other two registers having arbitrary values. It works by repeatatively going through a process cycle until a `halt` instruction is executed. A process cycle is:

1. Take $[\mathcal{I}]$ as an instruction.

2. If this instruction is the `halt` instruction then terminate the process,

3. else perform the action associated with that instruction.

4. Continue with step (1).

If any instruction described in the following two pages computes a result (at any cycle) that falls out of the limits then the machine automatically halts.

The HW-1 accepts 17 instructions which are explained below. Instructions are recognized as integers [0,1,...,16].

Instruction $\boxed{0}$

Halt the process..

Instruction $\boxed{1}$

Load $\mathcal{R}_1$ with the next number in the sequence.

$$\mathcal{R}_1 \leftarrow [\mathcal{I}+1]\,, \quad \mathcal{I} \leftarrow \mathcal{I}+2$$

Instruction $\boxed{2}$

Load $\mathcal{R}_2$ with the next number in the sequence.

$$\mathcal{R}_2 \leftarrow [\mathcal{I}+1]\,, \quad \mathcal{I} \leftarrow \mathcal{I}+2$$

Instruction $\boxed{3}$

Load $\mathcal{R}_1$ with the sequence element which is at the position given as the next number in the sequence.

$$\mathcal{R}_1 \leftarrow [[\mathcal{I}+1]]\,, \quad \mathcal{I} \leftarrow \mathcal{I}+2$$

Instruction $\boxed{4}$

Load $\mathcal{R}_2$ with the sequence element which is at the position given as the next number in the sequence.

$$\mathcal{R}_2 \leftarrow [[\mathcal{I}+1]]\,, \quad \mathcal{I} \leftarrow \mathcal{I}+2$$

Instruction $\boxed{5}$

Load $\mathcal{R}_1$ with the content of $\mathcal{R}_2$.

$$\mathcal{R}_1 \leftarrow \mathcal{R}_2\,, \quad \mathcal{I} \leftarrow \mathcal{I}+1$$

Instruction $\boxed{6}$

Load $\mathcal{R}_1$ with the sequence element which is at the position $\mathcal{R}_2$.

$$\mathcal{R}_1 \leftarrow [\mathcal{R}_2]\,, \quad \mathcal{I} \leftarrow \mathcal{I}+1$$

Instruction $\boxed{7}$

Change the sequence element which is at the position $\mathcal{R}_1$ to be the content of $\mathcal{R}_2$.

$$[\mathcal{R}_1] \leftarrow \mathcal{R}_2\,, \quad \mathcal{I} \leftarrow \mathcal{I}+1$$

Instruction $\boxed{8}$

Change the sequence element which is at the position given as the next number in the sequence to the content of $\mathcal{R}_1$.

$$[[\mathcal{I}+1]] \leftarrow \mathcal{R}_1\,, \quad \mathcal{I} \leftarrow \mathcal{I}+2$$

Instruction $\boxed{9}$

Take the sequence element which is at the position given as the next number in the sequence as the next instruction to be performed.

$$\mathcal{I} \leftarrow [\mathcal{I}+1]$$

Instruction $\boxed{10}$

If $\mathcal{R}_1$ contains zero continue with the sequence element following the next one as the next instruction to be performed, otherwise act like the instruction 9.

$$\begin{array}{lcl} \text{if } \mathcal{R}_1 = 0 & : & \mathcal{I} \leftarrow \mathcal{I}+2 \\ \text{otherwise} & : & \mathcal{I} \leftarrow [\mathcal{I}+1] \end{array}$$

Instruction $\boxed{11}$

Increment $\mathcal{R}_1$ by the content of $\mathcal{R}_2$.

$$\mathcal{R}_1 \leftarrow \mathcal{R}_1 + \mathcal{R}_2\ , \quad \mathcal{I} \leftarrow \mathcal{I}+1$$

Instruction $\boxed{12}$

Decrement $\mathcal{R}_1$ by the content of $\mathcal{R}_2$.

$$\mathcal{R}_1 \leftarrow \mathcal{R}_1 - \mathcal{R}_2\ , \quad \mathcal{I} \leftarrow \mathcal{I}+1$$

Instruction $\boxed{13}$

Multiply $\mathcal{R}_1$ by the content of $\mathcal{R}_2$.

$$\mathcal{R}_1 \leftarrow \mathcal{R}_1 \times \mathcal{R}_2\ , \quad \mathcal{I} \leftarrow \mathcal{I}+1$$

Instruction $\boxed{14}$

Divide $\mathcal{R}_1$ by the content of $\mathcal{R}_2$ (integer division).

$$\mathcal{R}_1 \leftarrow \mathcal{R}_1 \div \mathcal{R}_2\ , \quad \mathcal{I} \leftarrow \mathcal{I}+1$$

Instruction $\boxed{15}$

Change the sign of the value in $\mathcal{R}_1$.

$$\mathcal{R}_1 \leftarrow -\mathcal{R}_1\ , \quad \mathcal{I} \leftarrow \mathcal{I}+1$$

Instruction $\boxed{16}$

Compare the content of $\mathcal{R}_1$ with the content of $\mathcal{R}_2$.

$$\begin{array}{lcl} \text{if } \mathcal{R}_1 = \mathcal{R}_2 & : & \mathcal{R}_1 \leftarrow 0 \\ \text{if } \mathcal{R}_1 > \mathcal{R}_2 & : & \mathcal{R}_1 \leftarrow 1 \\ \text{otherwise} & : & \mathcal{R}_1 \leftarrow -1 \end{array}$$

$$\text{always } \mathcal{I} \leftarrow \mathcal{I}+1$$

An Example

If HW-1 is submitted the below given sequence, it will compute the sum of its 2nd and 3rd elements, then compare this sum with the 4th element. If the sum equals the 4th element then the 5th element of the sequence is changed to `1` else it is changed to `0`.

9	6	12	27	39	99	3	2	4	3	11	4	4	16	10	20	2	1	9	22	2	0	1	5	7	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

We will go through each cycle of the process and explain it:

Cycle 1

The machine starts now. So the registers are as follows:

$$\begin{array}{ccc} \mathcal{R}_1 & \mathcal{R}_2 & \mathcal{I} \\ \boxed{?} & \boxed{?} & \boxed{0} \end{array}$$

Since $\mathcal{I}$ contains 0, the 0th (zeroth) element of the sequence, $[\mathcal{I}]$ is fetched. This will be our current instruction. Looking at the sequence you may realize that this element is `9`. Look at the definition of the instruction 9 (given on the previous page), you will see that the action of the instruction 9 is:

$$\mathcal{I} \leftarrow [\mathcal{I}+1]$$

$\mathcal{I}+1$ is 1 and therefore $[\mathcal{I}+1]$ is simply $[1]$, in other words the 1st element of the sequence. That is `6`. So the value in $\mathcal{I}$ is changed by the assignment operation to be 6. And now the registers are:

$$\begin{array}{ccc} \mathcal{R}_1 & \mathcal{R}_2 & \mathcal{I} \\ \boxed{?} & \boxed{?} & \boxed{6} \end{array}$$

Cycle 2

$\mathcal{I}$ contains 6. The machine fetches $[6]$ which is `3`. Instruction 3 is defined as:

$$\mathcal{R}_1 \leftarrow [[\mathcal{I}+1]]\,, \quad \mathcal{I} \leftarrow \mathcal{I}+2$$

The first assignment will change the content of $\mathcal{R}_1$ to be $[[\mathcal{I}+1]]$. Now do not panic! $\mathcal{I}$ was holding a value of 6 therefore the inner brackets are nothing else but

$$[[\underbrace{\mathcal{I}+1}_{[6+1]}]] \qquad \text{which is} \quad [[7]]$$

$[7]$ is the 7th element of the sequence, namely `2`. So $[[7]]$ is nothing but $[2]$. As you must have got used to it by now, this is the 2nd element of the sequence, we go and fetch it. It is `12`. Since the first assignment was $\mathcal{R}_1 \leftarrow [[\mathcal{I}+1]]$ this value found on the right hand side of the assignment, namely the 12, will be stored into $\mathcal{R}_1$. The second assignment increments the value of $\mathcal{I}$ by 2. Thus $\mathcal{I}$ is now $6+2$ which is 8. At the end of this cycle the register contents are:

$$\begin{array}{ccc} \mathcal{R}_1 & \mathcal{R}_2 & \mathcal{I} \\ \boxed{12} & \boxed{?} & \boxed{8} \end{array}$$

Cycle 3

Since $\mathcal{I}$ is containing 8 we fetch [8] as the instruction of this cycle. The 8th element of the sequence is `4`. The instruction 4 is defined similar to the instruction 3 but now it is the $\mathcal{R}_2$ register that receives the value and not $\mathcal{R}_1$. Going through a very similar argumentation that we did for cycle 2, we conclude that $\mathcal{R}_2$ will be assigned the value [3] which is `27`. Following this assignment, $\mathcal{I}$ will be incremented by 2. That is how the registers look like:

$\mathcal{R}_1$	$\mathcal{R}_2$	$\mathcal{I}$
12	27	10

Cycle 4

This cycle's instruction is [10]. Looking at the sequence we see that this is the instruction `11`. An instruction which adds the content of $\mathcal{R}_2$ to the content of $\mathcal{R}_1$ and assigns the sum to $\mathcal{R}_1$*(Please go and check the definition of instruction* 11*)*. So the following operation is performed:

$$\mathcal{R}_1 \leftarrow 12 + 27$$

So, $\mathcal{R}_1$ is assigned a value of 39. Due to the definition of instruction 11, the $\mathcal{I}$ register is incremented by 1. At the end of this cycle the registers read as:

$\mathcal{R}_1$	$\mathcal{R}_2$	$\mathcal{I}$
39	27	11

Cycle 5

Our instruction is [11] which is `4`. We have had a similar instruction in cycle 3. But this time $[\mathcal{I}+1]$ is `4` which means that $\mathcal{R}_2$ is assigned the value [4]. From the sequence we get that this value is 39. So $\mathcal{R}_2 \leftarrow 39$. By the definition, $\mathcal{I}$ is incremented by 2. Now the registers are:

$\mathcal{R}_1$	$\mathcal{R}_2$	$\mathcal{I}$
39	39	13

Cycle 6

The instruction of this cycle is [13], namely `16`. This is a comparison test performed between $\mathcal{R}_1$ and $\mathcal{R}_2$. If they are equal $\mathcal{R}_1$ is changed to 0, if the value of $\mathcal{R}_1$ is greater than the value of $\mathcal{R}_2$ then $\mathcal{R}_1$ is changed to 1. If it is the third possibility, which is the only one left, namely the case where the value of $\mathcal{R}_1$ is lesser than the value of $\mathcal{R}_2$ then $\mathcal{R}_1$ is changed to -1. In all cases $\mathcal{I}$ is incremented by one. Following this definition and looking at the contents of the registers we can conclude that HW-1 will change $\mathcal{R}_1$ to 0 since both $\mathcal{R}_1$ and $\mathcal{R}_2$ have the value 39, hence they are equal. The registers are:

$\mathcal{R}_1$	$\mathcal{R}_2$	$\mathcal{I}$
0	39	14

Cycle 7

The instruction is [14] which reads as `10`. This is a conditional jump *('jump on non-zero')*. If the register $\mathcal{R}_1$ contains a non zero value then the instruction of the next cycle will be fetched from the sequence position that given at $[\mathcal{I}+1]$. In our case the register $\mathcal{R}_1$ indeed contains a 0. So the jump will not take place and the $\mathcal{I}$ register will simply be incremented by 2. *(If the jump would have taken place then $\mathcal{I}$ would be set to `20` and this would be the position in the sequence from which the instruction for cycle 8 would be fetched)* So the registers are

$\mathcal{R}_1$	$\mathcal{R}_2$	$\mathcal{I}$
0	39	16

Cycle 8

The instruction is now [16], that is `2`. This will load $\mathcal{R}_2$ with $[\mathcal{I}+1]$. In our case this is $\mathcal{R}_2 \leftarrow [17]$ and will result in $\mathcal{R}_2 \leftarrow 1$. For an instruction 2, $\mathcal{I}$ will be incremented by 2. At the end of this cycle we have in the registers

$\mathcal{R}_1$	$\mathcal{R}_2$	$\mathcal{I}$
0	1	18

Cycle 9

The instruction is [18] which is `9` an instruction that performs an unconditional jump. $\mathcal{I}$ will be set to $[\mathcal{I}+1]$ and that is `22`. The registers are

$\mathcal{R}_1$	$\mathcal{R}_2$	$\mathcal{I}$
0	1	22

Cycle 10

The instruction is [22] which is `1`. This instruction will load $\mathcal{R}_1$ with $[\mathcal{I}+1]$. For our case this is `5`. Instruction 1 increments $\mathcal{I}$ by 2. Now we have the registers as

$\mathcal{R}_1$	$\mathcal{R}_2$	$\mathcal{I}$
5	1	24

Cycle 11

We are almost at the end! The instruction in turn is [24], that is `7`. An instruction that performs

$$[\mathcal{R}_1] \leftarrow \mathcal{R}_2$$

and then increment $\mathcal{I}$ by one. Due to this definition a $[5] \leftarrow 1$ is carried out. And for the first time we have changed a value in the sequence. Now the 5th element is no more `99` but 1. This was the claimed action of the HW-1 machine with this example sequence. After a following incrementation of $\mathcal{I}$ the registers are:

$\mathcal{R}_1$	$\mathcal{R}_2$	$\mathcal{I}$
5	1	25

Cycle 12

Yes, unbeliveable but true, the machine stops and we all go home! The instruction for this cycle is [25] which is simply `0`. And this means `halt`.

the final picture of the sequence is:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
9	**6**	**12**	**27**	**39**	**1**	**3**	**2**	**4**	**3**	**11**	**4**	**4**	**16**	**10**	**20**	**2**	**1**	**9**	**22**	**2**	**0**	**1**	**5**	**7**	**0**

For sake of completeness we display the registers at the moment of the `halt`:

$\mathcal{R}_1$	$\mathcal{R}_2$	$\mathcal{I}$
5	1	25

"The name of the song is called 'Haddocks' Eyes.'"
"Oh, that's the name of the song, is it?" Alice said trying to feel interested.

"No, you don't understand," the Knight said, looking a little vexed.
"That's what the name is ***called****. Then name really* ***is*** *'The Aged Aged Man.'"*
"Then I ought to have said 'That's what the ***song*** *is called'?" Alice corrected herself.*

"No, you oughtn't: that's quite another thing! The ***song*** *is called 'Ways and Means': but that's only what it's* ***called****, you know!"*
"Well, what ***is*** *the song, then?" said Alice, who was by this time completely bewildered.*

"I was comming to that," the Knight said. "The song really is 'A-sitting On A Gate': and the tune's my own invention."

–Lewis Carroll, Through the Looking Glass