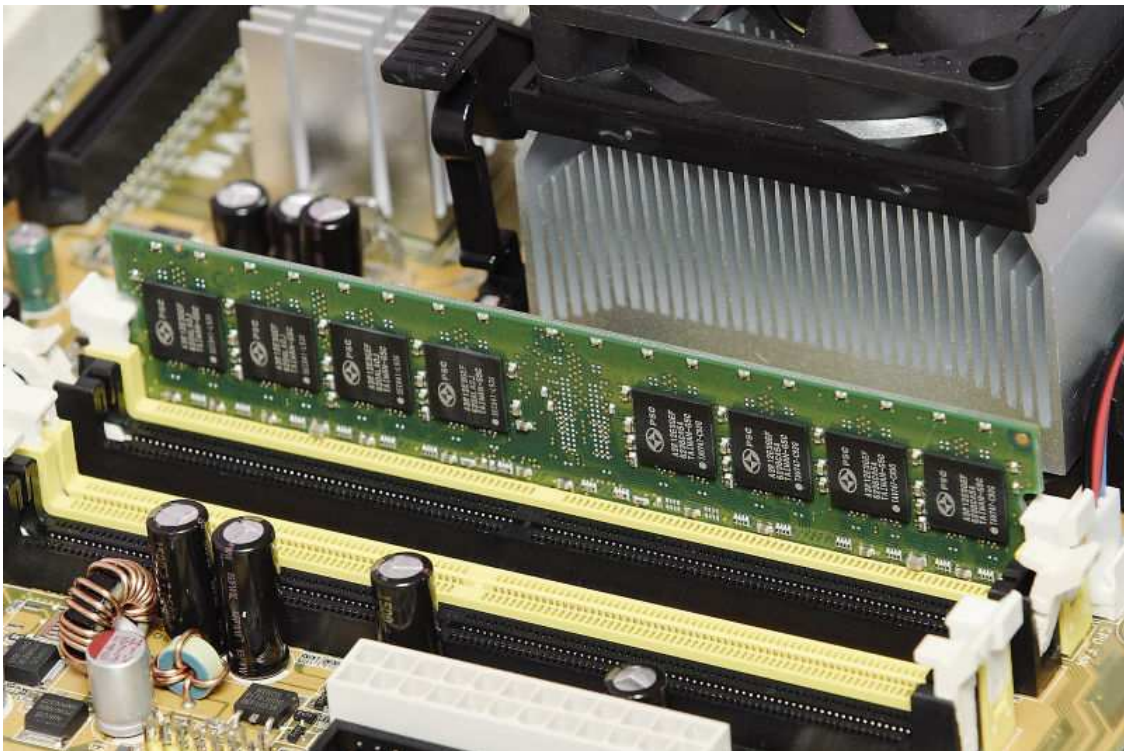


## The human computer Role-sheet: memory

**You're the memory.** A computer's memory can be viewed as a list of cells into which numbers can be placed or read. Each cell has a numbered "address" and can store a single number. The computer can be instructed to "put the number 123 into the cell numbered 1357" or to "add the number that is in cell 1357 to the number that is in cell 2468 and put the answer into cell 1595". The information stored in memory may represent practically anything: letters, numbers, but also (this is very important) computer instructions. Since the CPU does not differentiate between different types of information, it is up to the software to give significance to what the memory sees as nothing but a series of numbers.

In almost all modern computers, each memory cell is set up to store binary numbers in groups of eight bits (called a byte). Each byte is able to represent 256 different numbers; either from 0 to 255 or -128 to +127. To store larger numbers, several consecutive bytes may be used (typically, two, four or eight). Modern computers have billions or even trillions of bytes of memory.



## Preparation

The memory in our human computer is composed of 8 cells, each with an address. Each cell is role-played by one person (hence a total of 8 persons).

1. Each of the 8 persons chooses one address in the following list:

1111 0000 (Fo)	1111 0001 (F1)	1111 1010 (FA)	1111 1011 (FB)
1111 1100 (FC)	1111 1101 (FD)	1111 1110 (FE)	1111 1111 (FF)

Each person should remember its address very carefully. When a request comes to the memory (from other parts of the computer), it will refer to a specific cell by its address. The memory (i.e. the 8 persons composing it) should be able to react very quickly to requests like: “what is the value contained by cell FB?”

2. Each of the 8 persons works out and writes down the hexadecimal value corresponding to its initial value given in the table below.

Each letter in the instruction names should be coded as a byte, i.e. 2 hexadecimal digits. Each numerical values should be coded as a byte too, i.e. 2 hexadecimal digits. For instance, number “1” should be coded as “01” (i.e. the hexadecimal value corresponding to the binary representation “0000 0001”)

Address	Value
Fo	13
F1	7
FA	MOV 0, Fo
FB	MOV 1, F1
FC	ADD F1, Fo
FD	ADD 1, F1
FE	CMP F1, 10
FF	BLE FC

3. Collectively, double check that everyone has the correct hexadecimal value. Any mistake at this point will result in a failure at the global level for the whole class. Lesson learned: the reliability of memory is crucial. A single deficient cell can ruin the whole process.

4. Modifications will be made in the memory by other parts of the computer. For instance, the value in Fo may be changed to value "oB". Each cell/person is responsible to keep track of her own value. Write your current value on a piece of paper, and be ready to erase it and replace by a new value at any time.
  
5. Organize collectively so that operations like reading a value from a given address or changing the value at a given address can be made at maximum speed and maximum reliability.