Network register I.

Submission deadline:	**2019-04-28 23:59:59**
Late submission with malus:	**2019-06-30 23:59:59** (Late submission malus: 100.0000 %)
Evaluation:	**4.0000**
Max. assessment:	**4.0000** (Without bonus points)
Submissions:	3 / 20 Free retries + 20 Penalized retries (-2 % penalty each retry)
Advices:	0 / 2 Advices for free + 2 Advices with a penalty (-10 % penalty each advice)

The problem is to design and implement classes that simulate a database of networked computers. We need to store information about networks (`CNetwork`), computers (`CComputer`) and their components: `CCPU`, `CMemory`, and `CDisk`.

This assignment is focused on class design, where inheritance, polymorphism and abstract methods are used. If these OOP paradigms are used correctly, the implementation is short and clean. On the other hand, if the design is wrong, the implementation will be lengthy with repeated code. Try to identify base class and subclasses, use inheritance.

The classes and the interface:

`CNetwork`
: represents a network. The interface is:

- constructor with the network name parameter,
- destructor, copy constructor and operator = (if the automatically generated are not correct),
- method `AddComputer` which adds another computer to the list,
- method `FindComputer` which returns a pointer to `CComputer` object with the given name, or an empty pointer if the computer of that name does not exist,
- output operator which displays the network in the format shown below. The computers are listed in the order they were added into the network object.

`CComputer`
: represents a computer. The interface is:

- constructor with computer name parameter (string),
- destructor, copy constructor and operator = (if the automatically generated are not correct),
- method `AddComponent` which adds another component to the list,
- method `AddAddress` which adds another address to the list of addresses,
- output operator which displays the computer in the format shown below. The addresses are listed first (in the order they were added), followed by the list of components (again in the order they were added).

`CCPU`
: represents a CPU. The interface is:

- constructor with the number of cores (int) and frequency (int, MHz) parameters,
- destructor, copy constructor and operator = (if the automatically generated are not correct).

`CMemory`
: represents a RAM memory. The interface is:

- constructor with the memory size (int, in MiB),
- destructor, copy constructor and operator = (if the automatically generated are not correct).

`CDisk`
: represents a storage. The interface is:

- constructor with the storage type (symbolic constant `SSD` or `MAGNETIC`) and disk size (int, in GiB),
- destructor, copy constructor and operator = (if the automatically generated are not correct),
- method `AddPartition` which adds another partition to the disk description. The method will take two parameters: partition size (int, in GiB) and a partition description (string). The partitions are listed in the order they were added.

Submit a source code with the implementation of classes `CNetwork`, `CComputer`, `CCPU`, `CMemory`, and `CDisk`. All required auxiliary declarations/functions shall be included in the source file submitted. The #include preprocessor definitions and your tests shall be placed in the conditional compile blocks (as in the attached sample).