Lab 02 - IAS Computer

Instructions:

- The IAS is one of the original stored-program computers. It has **21 opcodes** (operation codes), namely

Mnemonic	Opcode	Description
LMA	0A	Transfer contents from MQ to AC
LDM	09	Transfer M(X) to MQ
STA	21	Transfer contents from AC to memory location X
LDA	01	Transfer M(X) to AC
LDN	02	Transfer -M(X) to AC
ALD	03	Transfer |M(X)| to AC
ALN	04	Transfer -|M(X)| to AC
BRL	0D	Takes next instruction from left half of M(X)
BRR	0E	Takes next instruction from right half of M(X)
BPL	0F	If AC $>= 0$, takes next instruction from the left half of M(X)
BPR	10	If AC $>= 0$, takes next instruction from the right half of M(X)
ADD	05	Add M(X) to AC; put result in AC
AAD	07	Add |M(X)| to AC; put result in AC
SUB	06	Subtract M(X) from AC; put result in AC
ASB	08	Subtract |M(X)| from AC; put result in AC
MUL	0B	Multiply M(X) by MQ; put most significant bits of result in AC; least significant in MQ
DIV	0C	Divide AC by M(X); put quotient in MQ and remainder in AC
LSH	14	Multiply AC by 2
RSH	15	Divide AC by 2
STL	12	Transfer AC[28:39] to M(X)[8:19]
STR	13	Transfer AC[28:39] to M(X)[28:39]
HLT	00	Halts

Your objective is to complete Note01, simulate the execution of an IAS program by tracing the changes to its registers and memory.

- Each source code written must compile and include only the libraries 'iostream', 'iomanip', 'string', 'sstream', 'fstream', 'cctype', 'cmath', 'stdexcept', 'Memory.h', 'IAS.h' and 'Object.h' from Note01, and user-defined libraries from the lab to receive any credit

- A cumulative task will not receive credit if the required previous tasks are not completed.

- Your submissions must be submitted to the GitHub repository in the Lab02 directory.

- Cheating of any kind is prohibited and will not be tolerated.

- **Violating or failing to follow any of the rules above will result in an automatic zero (0) for the lab.**

Grading

Task	Maximum Points	Points Earned
1	2.4	
2	1.4	
3	1.2	
Total	5.0	

Note: solutions will be provided for tasks colored blue only.

Task 1

- Create a header file 'Decode.h' that defines the class *Decode* in the namespace *osl* that contains

  - a public void static method named `execute()` that takes an *IAS* reference parameter, and decodes and performs the opcode stored in the IR register of the *IAS* parameter.
  - a private void static method that takes an *IAS* reference parameter for each IAS operation.

Task 2

- Create a header file 'Import.h' that defines within the namespace *osl* the void function named `Import()` that takes an *IAS* reference and a string as parameters. If the file specified by the string parameter opens successfully, the function shall read the file line by line and sequentially store each valid value into consecutive memory cells of the provided *IAS* object. The function must continue reading and storing values until one of the following conditions occurs:

  - A line does not represent a valid hexadecimal number containing at most 10 digits, or
  - A maximum of 4096 lines have been read from the file.

  Furthermore, every member cell assigned a zero must be deactivated.

Task 3

- Create a C++ file named 'main.cpp' that defines a void function named `simulate()` that takes an *IAS* reference parameter and an ofstream reference parameter. The simulate function shall execute the complete instruction cycle of the provided *IAS* object. It must repeatedly perform the fetch–decode–execute cycle as defined in Note01, continuing execution until a halt instruction is encountered. During execution:

  - The function shall process each fetch cycle step exactly as specified in Note01.
  - After every assignment and execute operation within the cycle, the current state of the IAS object shall be displayed.
  - Each display output shall be written to the file associated with the provided std::ofstream object.