Project 1: Float Number

Part 1: Implement a class called FloatNumber.
a) A class named FloatNumber
b) Private members integer (long long for larger numbers) and decimal (long long for precision).
c) Accessors getInteger(), getDecimal() and mutators setInteger(long long), setDecimal(long long).

d) Write a part in main function which test all function and class component implemented in this part.

Part 2: Continue with FloatNumber class:
e) Default constructor setting both integer and decimal to 0, Implement constructor delegation
f) One-parameter constructor (double) which splits the number into integer and decimal parts.
g) Two-parameter constructor (long long, long long) for setting integer and decimal directly.
h) Write a part in main function which test all function and class component implemented in this part.

Part 3: Continue with FloatNumber class:
i) Overload the insertion operator << to output FloatNumber objects.
j) Overload the extraction operator >> to input FloatNumber objects.
k) Write a part in main function which test all function and class component implemented in this part.

Part 4: Continue with FloatNumber class:
l) Overload the operator +, -, *, / to output FloatNumber objects, with auto conversion.
For division, consider throwing an exception for division by zero or precision loss.
m) Write a part in main function which test all function and class component implemented in this part.

Part 5: Continue with FloatNumber class:
n) Overload the operator >, >=, <, <=, ==, != to output boolean, with auto conversion.
o) Write a part in main function which test all function and class component implemented in this part.

Part 6: Continue with FloatNumber class:
p) Overload the assignment operators +=, -=, /=, *=
Ensure these operations update in place while maintaining precision.

q) Write a main function which test all function and class component implemented in this part.

Part 7: Continue with FloatNumber class:
r) Overload the subscript [] operator so that index 0 returns integer part and index 1 returns decimal part.
s) Write a main function which test all function and class component implemented in this part.

Part 8: Continue with FloatNumber class:

t) Overload the prefix and postfix ++ operators, so they return a FloatNumber object with value increased by one.
u) Overload the prefix and postfix - - operators, so they return a FloatNumber object with value decreased by one.
v) Write a main function which test all function and class component implemented in this part.

Part 9: Implement separate compilation, and create a makefile for the project. Move class definition and implementations into separate .h and .cpp files. Write a Makefile to compile the project.