You are to write a C++ program with several classes and instances of those classes.

The principal class will be called Animal, and it will have four different subclasses for four different types of animal (such as Cow, Horse, Cat, Dog, Mouse, Sheep, etc.). The choice of these subclasses is left to you.

Animal, and each of its subclasses, should have member functions:

```cpp
type name
string noise() // returns a string for the noise of the animal.
int repeats() // returns the number of times the animal likes to repeat its noise.
int ageInMonths() // returns the number of months this animal has been alive.
void print()
```

You choose the noise string for each animal, for instance, a Cow might have a noise of “moo”, and an Animal might have a noise of “grunt”. The noise should be constant for all instances of a particular class (all Cows should say “moo”). Do not use a variable to store the noise; use a string constant in the noise function. Make noise() have dynamic binding.

Similarly, each type of animal has its own favored number of repeats between 1 and 5. (For example, all Cows might like to moo twice, and all Animals might like to grunt three times.) Again, do not use a variable to store the repeats number; use an integer constant in the repeats function. Make repeats() have static binding.

Each instance of an animal or its subclasses will have its own age. This should be passed to the animal via a constructor argument, and stored in a private member variable. Implement ageInMonths() for Animal and let your subclasses inherit it.

Similarly implement print() for Animal and not for the subclasses. Print should print the animal’s age, followed by its noise() repeated repeats() times, all on a single line, followed by a newline. For instance, a sheep of 8 months age might print:

```
8 baa baa baa
```
In your main() subroutine (or some other subroutine that is called from main()), create a variable (instance) of each class that you have defined (five variables total for this). Then run print() for each of these variables, and print a blank line afterwards.

After that, declare an array of 10 pointers to Animals, and initialize each one with all the different subclasses and an instance of the superclass itself. For instance:

```cpp
animals[0] = new Cat ...
animals[1] = new Mouse ...
animals[2] = new Cow ...
animals[3] = new Animal ...
animals[4] = new Mouse ...
...
```

After this initialization, loop through all the elements of the array, executing print() on each one. Because repeats() is statically bound, each animal should repeat its noise the same number of times.

At the end of the program, delete all of the different animals that you allocated with `new`.

That’s it for this assignment; it’s intentionally a straightforward one so that you can get used to the compiler and the submission system and because there’s only a week or so to get it done. Future assignments will also relate to lecture but may require deeper implementation.

Submit your .h and .cpp files in a zip (preferable) or tar archive. Don’t submit your .exe or .o files.

You will be judged on correctness of your code and on code style, so don’t forget to keep your code clean as you develop it! (Or at the very least, clean it up before submission. We don’t want to see untidy code.)