1) Since it’s private, the main function cannot directly call it. It needs to have a public function which would then call the private function, sort of like how a getter functions.

2) You end up defining a function which doesn’t exist

3) The Pet class expects two arguments since our only constructor requires two arguments. If we had created a constructor which would instantiate a default value for health, then there wouldn’t (shouldn’t) be an issue.

4) An error stating the prototype for ‘bool Pet::Alive()’ doesn’t match anything in the Pet class

5) Well, you end up calling a a function which isn’t const, on a const object. And an insightful StackOverflow comment points out that “...calling a non-const member function on [a] const object which is not allowed because non-const member functions make NO PROMISE not to modify the object.” So, the compiler prevents any issues of data manipulation by the calling function on something which was passed in exclusively as a const object. It works in careFor/main simply because the Pet object isn’t classified as a const in the function argument.