**C# Homework 07**

**Question 1**

What is a class? According to the book, what does a class “arrange”?

**Answer**

Class is the root word of the term classification. When you design a class, you systematically arrange information and behavior into a meaningful entity. This arranging is an act of classification and is something that everyone does, not just programmers.

**Question 2**

What are the two purposes of encapsulation?

**Answer**

It’s two purposes are: to combine methods and data within a class; in other words, to support classification, and to control the accessibility of the methods and data; in other words, to control the use of the class.

**Question 3**

How do you instantiate an instance of a class? How do you access that instance?

**Answer**

Specify the name of the class and declare a variable. To access it you call it.

**Question 4**

What is the default access of the fields and methods of a class? How do you change the default?

**Answer**

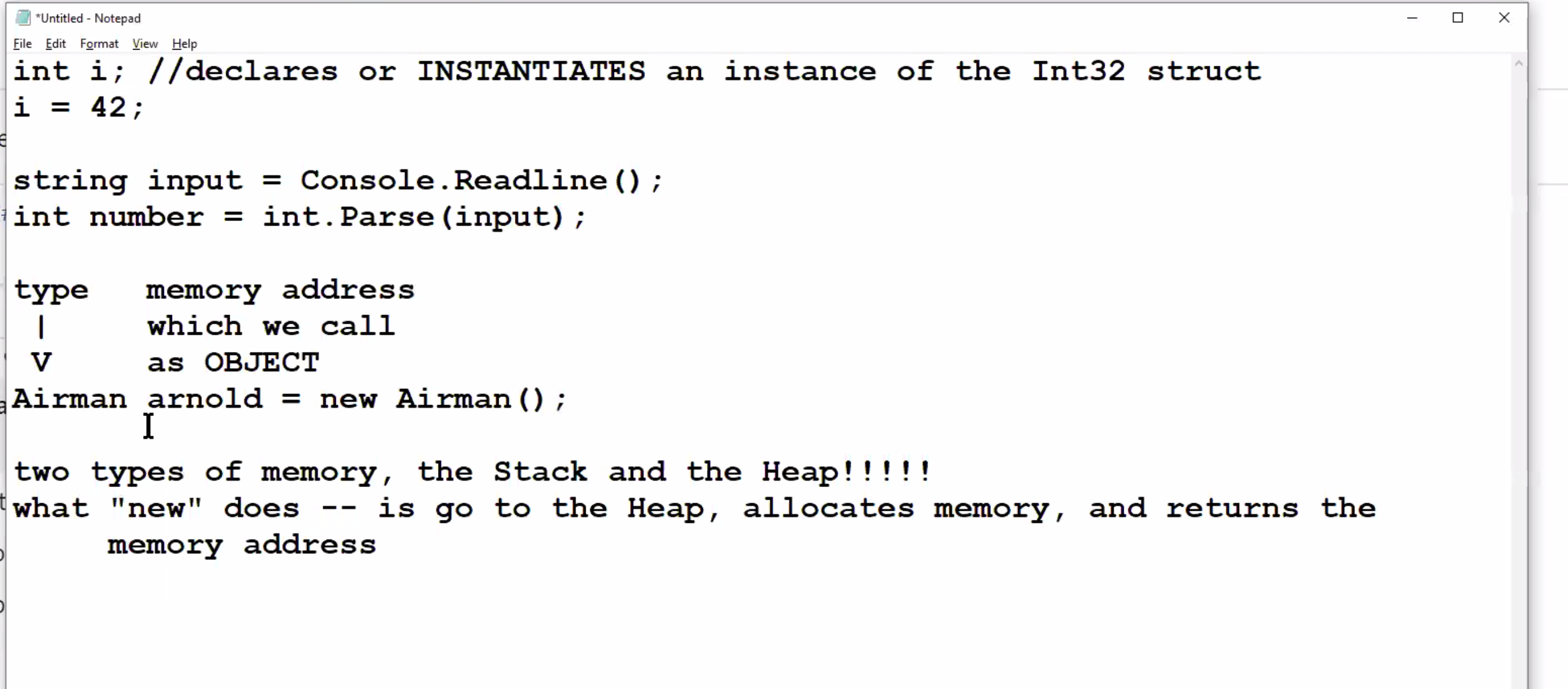
A method or field is private if it is accessible only from within the class. To declare that a method or field is private, you write the keyword private before its declaration. As intimated previously, this is actually the default, but it is good practice to state explicitly that fields and methods are private to avoid any confusion. A method or field is public if it is accessible both within and from outside the class. To declare that a method or field is public, you write the keyword public before its declaration.

**Question 5**

What is the syntax for writing a constructor?

**Answer**

A constructor is a special method that runs automatically when you create an instance of a class. It has the same name as the class, and it can take parameters, but it cannot return a value (not even void). Every class must have a constructor. If you don’t write one, the compiler automatically generates a default constructor for you. You can write your own default constructor quite easily. Just add a public method that does not return a value and give it the same name as the class.



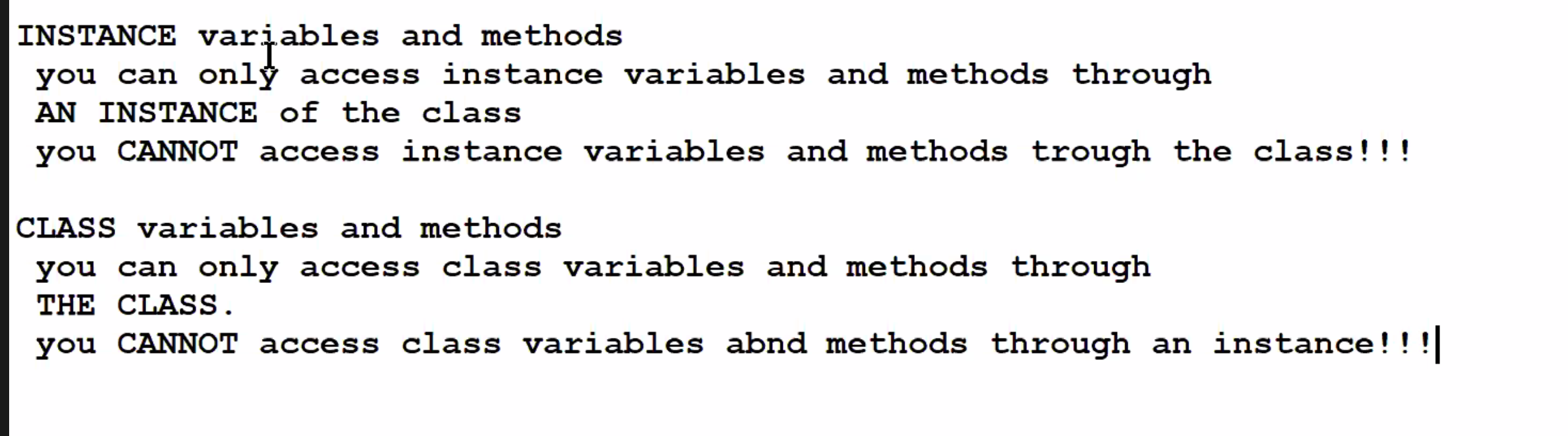
**Question 6**

What is the difference between class fields and methods, and instance fields and methods? How do you

create class fields and methods?

**Answer**

Instance methods can access class variables and class methods directly. Class methods can access class variables and class methods directly. Class methods cannot access instance variables or instance methods directly-they must use an object reference.



**Question 7**

How do you bring a static class in scope? Why would you want to bring a static class in scope?

**Answer**

The advantage of using a static class is that the compiler can check to make sure that no instance members are accidentally added. The compiler will guarantee that instances of this class cannot be created. Static classes are sealed and therefore cannot be inherited. They cannot inherit from any class except Object.

**Question 8**

Can you think of a good reason to create an anonymous class? What is it?

**Answer**

An anonymous class is a class that does not have a name. It is handy when using query expressions.

**Question 9**

What is polymorphism as this term is used in computer science? This is not in the book.

**Answer**

Polymorphism refers to the ability to present the same interface for different forms. Although the concept of polymorphism is the same in all programming languages that support it, its implementation differs from one language to another. Operator overloading is an example of this type of polymorphism.

**Question 10**

What is message passing as this term is used in computer science? This is not in the book.

**Answer**

Use message passing to distinguish it from the imperative notion of “calling a function”, and to reinforce the idea that the receiving object decides what to do. An example would be methods that are abstract in a class and implemented in subclasses; or implementation of an interface method.

**Question 11**

What was the first object-oriented programming language?

**Answer**

Simula

**Question 12**

Consider this quote by Alexander Stephanov:

I find OOP technically unsound. It attempts to decompose the world in terms of interfaces that vary on a single type. To deal with the real problems you need multi-sorted algebras – families of interfaces that span multiple types. I find OOP philosophically unsound. It claims that everything is an object. Even if it is true it is not very interesting – saying that everything is an object is saying nothing at all.

Who is Alexander Stephanov? What do you think about this quote?

**Answer**

Alexander Stephanov is a Russian-American computer programmer, best known as an advocate of generic programming and as the primary designer and implementer of the C++ Standard Template Library. He does not like object oriented programming.