

SIT771 Object Oriented Development

Pass Task 1.3: How Many Objects

Overview

One challenge with object oriented programming is understanding the differences between classes, objects, and variables. Activity 3 of Course 1 has details that explain these concepts.

In this tasks you will need to read supplied code, and answer some questions about the number of objects and variables used. This will help you develop your code reading and analysis skills.

For this task you need to read the supplied code and answer the associated questions.

Submission Details

Submit the following files to Doubtfire.

- Your answers as a PDF

You want to focus on having clear explanations that demonstrate the differences between classes, objects, and variables.

Instructions

In C#, classes are used to define the what objects created from that *class* will know and be able to do. You can then use the *class* to create **new objects** when your program runs. To keep track of these objects we need to use *variables*, which keep track of the objects we want to remember in our code.

There are two main kinds of variables in C#, value and reference variables. We will look at this in detail in Course 3 and Course 5, but for the moment it is important that you picture variables as **referring** to objects. The following picture illustrates the difference between a value variable (like `count` that stores a number) and a reference variable (like `p` which *refers* to a Person object). The value in the `p` variable stores the memory location of the `Person` object it refers to, though it is best to just picture this as a **pointer** (arrow) that says where to find the Person object.

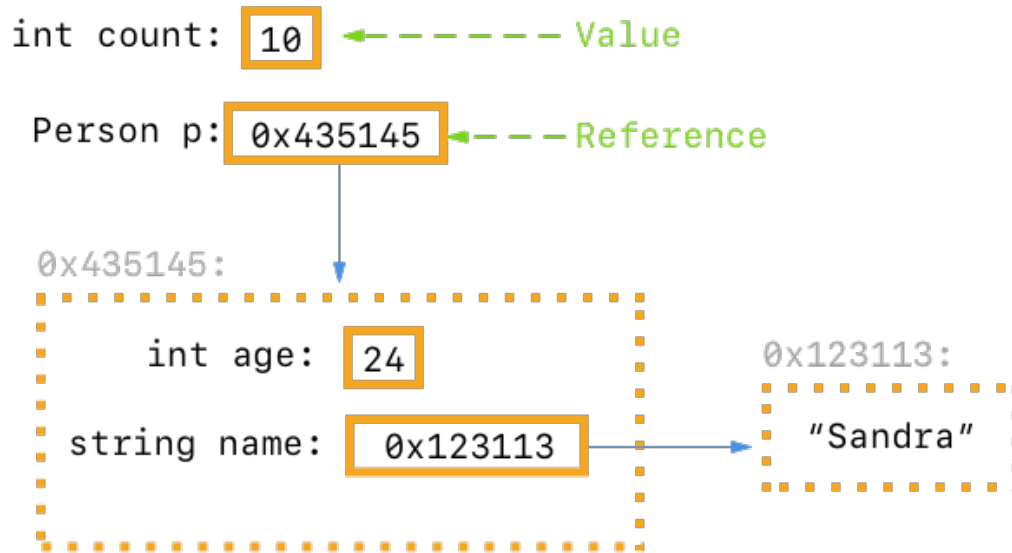


Figure: Variables store references to objects, they are not the objects themselves.

Here are a few points to remember when reading the following code and answering the questions:

- Variables only ever **refer** to objects in C#
- This means you can have *many variables* referring to *one* object.
- On its own, declaring a variable does not create an object.
- Objects are only ever created using the **new** keyword.

With this in mind, read the following code, and then answer the questions in the *How Many Objects.docx* file, which you can download from Doubtfire.

Code for Questions

```
Window helloWindow;
Window anotherWindow;
Window yetAnotherWindow;

helloWindow = new Window("Hello World", 800, 600);
anotherWindow = new Window("Another Window", 300, 300);
yetAnotherWindow = helloWindow;

helloWindow.MoveTo(0, 0);

anotherWindow.Clear(Color.Green);
anotherWindow.Refresh(60);

yetAnotherWindow.Clear(Color.Blue);
yetAnotherWindow.Refresh(60);

Bitmap pegasi = new Bitmap("Pegasi", "Pegasi.png");

helloWindow.DrawBitmap(pegasi, 10, 50);
anotherWindow.DrawBitmap(pegasi, 50, 10);

helloWindow.Refresh(60);
anotherWindow.Refresh(60);

SoundEffect knock = new SoundEffect("Knock", "Knocking-84643603.wav");
knock.Play();

SplashKit.Delay(5000);
```

Here are some helpful hints.

- The `Clear` method will change color of the window. The default color is `Grey` – so if the window isn't told to clear it will have a Grey color.
- You tell an object to do something by calling a method on the object. For example `helloWindow.MoveTo(0, 0)` tells the object referred to by the `helloWindow` variable to move to a new location.