Matthew Lonis

Jeff Whitmer

CSCI-A 290

October 28th, 2016

***MINI 2***

**Topic 1: Variable**

**3 URLs**

1. <https://launchschool.com/books/ruby/read/variables>
2. <http://www.cs.utah.edu/~germain/PPS/Topics/variables.html>
3. <http://www.makeuseof.com/tag/basics-of-computer-programming-variables-datatypes/>

**Precis of Sources**

Variables are where the dynamic information of a program is stored. Variables are not all the same though and are usually associated with a certain data type like integer or character for example. Variables are symbolic and are essentially names that act like placeholders for data. Variables only have a few things that you can do with them. You can create a variable, you can put information into or change information in a variable, or you can get a copy of the information out of the variable. There are six properties associated with a variable. These are: A name, type, value, scope, life time, and location in memory. The name is Symbolic. It represents the "title" of the information that is being stored with the variable. Every variable must have a unique name! The type represents what "kind" of data is stored with the variable. (See the Chapter on Data Types). A variable, by its very name, changes over time.

Good programs are "Chopped" into small self-contained sections (called functions) much like a good novel is broken into chapters, and a good chapter is broken into paragraphs, etc. A variable that is seen and used in one function is NOT available in another section. This allows us to reuse variable names, such as age. In one function 'age' could refer to the age of a student, and in another function 'age' could refer to the vintage of a fine wine. Further this prevents us from "accidentally" changing information that is important to another part of our program. The life time of a variable is strongly related to the scope of the variable. When a program begins, variables "come to life" when the program reaches the line of code where they are "declared". Variables "die" when the program leaves the "Scope" of the variable. Luckily, we don't have to worry too much about where in the computer hardware the variable is stored. The computer does this for us.

**Summary of Topic**

The two most important aspects of variables are in how they are interpreted and what they can represent. Variables can be interpreted hard or soft. Now what this means is that if a variable is interpreted softly, a program could add strings and integers, even though the data types aren’t compatible. Hard programming is referring to a programing language or compiler that does not allow a variable to do or perform operations that are outside of what it can do. The other important part about variables is that they can literally represent something as simple as a number or it can represent an entire structure.

**JIT 2**

Is C# a hard or soft interpreted language? In other words, is C# strictly typed or no?

**Topic 2: Type**

**3 URLs**

1. <http://study.com/academy/lesson/data-types-in-programming-numbers-strings-and-others.html>
2. <https://www.tutorialspoint.com/computer_programming/computer_programming_data_types.htm>
3. <http://www.cs.utah.edu/~germain/PPS/Topics/data_types.html>

**Precis of Sources**

Data types are very important in programming, especially for variables. As its name indicates, a data type represents a type of the data which you can process using your computer program. It can be numeric, alphanumeric, decimal, etc. Some common primitive data types include numbers (integers, doubles, etc.), characters and strings, arrays, and Booleans. More advanced data types exist and are called composite data types.

These data types include things like strings (in C), structures, and lists for starters. It’s important to note these data types are often made up by or contain primitive data types. Strings in C for example are made by making a character array. Some programming languages like Java include strings as a primitive data type. The age of the C programming language shows here. C doesn’t even include Boolean data types.

**Summary of Topic**

Going back to variables, data types are important in the sense that a programmer needs to be aware of what programming environment he or she is in. Depending on the environment, and what the programmer wants, careful steps need to be taken in order to not cause errors by manipulating variables outside of their intended functions.

**JIT 2**

Does C# have floats and does it have long and short prefixes for numbers?

**Topic 3: Procedure**

**3 URLs**

1. <https://www.techopedia.com/definition/3727/procedure>
2. <https://en.wikibooks.org/wiki/Computer_Programming/Procedures_and_functions>
3. <http://www.computerhope.com/jargon/p/procedure.htm>

**Precis of Sources**

In computer programming, a procedure is an independent code module that fulfills some concrete task and is referenced within a larger body of source code. This kind of code item can also be called a function or a sub-routine. The fundamental role of a procedure is to offer a single point of reference for some small goal or task that the developer or programmer can trigger by invoking the procedure itself.

In general programming theory, when an institution using the programs can have libraries of millions of programs collectively with billions of lines of source code, we want to avoid duplication of having the same code in multiple places. Suppose something needs changing. There is a new error message to add to the collection. Customer numbers just went from being 6 digit #s to 8 digits. There is a new currency, such as the Euro, to be handled. If this stuff was referenced in every single program that worked with it, we could have tens of thousands of programs to have to update to handle the change. But if there are sub-programs to handle different scenarios, then when that scenario changes, the only update is to the relevant sub-program.

**Summary of Topic**

The most important thing about procedures is that they are not related to an object or a class and are typically used in procedural programming. Procedures can be thought of much like function and much like methods, however procedures are different than methods because they are not object based.

**JIT 2**

Do procedures exist inside object oriented programs? In other words, can a programmer make a procedure that isn’t associated with an object and use it in an objectively based program?

**Topic 4: Method**

**3 URLs**

1. <http://whatis.techtarget.com/definition/method>
2. <http://study.com/academy/lesson/oop-object-oriented-programming-objects-classes-interfaces.html>
3. <http://stackoverflow.com/questions/155609/difference-between-a-method-and-a-function>

**Precis of Sources**

In object-oriented programming, a method is a programmed procedure that is defined as part of a class and included in any object of that class. A class (and thus an object) can have more than one method. A method in an object can only have access to the data known to that object, which ensures data integrity among the set of objects in an application. A method can be re-used in multiple objects.

A function is a piece of code that is called by name. It can be passed data to operate on (i.e. the parameters) and can optionally return data (the return value). All data that is passed to a function is explicitly passed. A method is a piece of code that is called by a name that is associated with an object. In most respects, it is identical to a function except for two key differences: A method is implicitly passed the object on which it was called and a method can operate on data that is contained within the class (remembering that an object is an instance of a class - the class is the definition, the object is an instance of that data).

**Summary of Topic**

Methods are much like procedures except that they are tied to an object or a class. Methods are nice inside objects and classes because it allows for programs to be divided up and to organize methods into classes that are easily understandable. Another good point I learned is that object based programming came out of the need to reuse code and reuse procedures.

**JIT 2**

In C#, are methods going to work much like how methods work in Java?

**Topic 5: Event**

**3 URLs**

1. <https://en.wikipedia.org/wiki/Event-driven_programming>
2. <http://www.computerhope.com/jargon/e/event.htm>
3. <http://www.webopedia.com/TERM/E/event.html>

**Precis of Sources**

In programming, an event is an action that occurs because of the user or another source, such as a mouse being clicked, or a key being pressed. An event handler is a routine that is used to deal with the event, allowing a programmer to write code that will be executed when the event occurs.

In computer programming, Event-Driven programming is a programming paradigm in which the flow of the program is determined by events such as user actions (mouse clicks, key presses), sensor outputs, or messages from other programs/threads. Event-driven programming is the dominant paradigm used in graphical user interfaces and other applications (e.g. JavaScript web applications) that are centered on performing certain actions in response to user input.

**Summary of Topic**

An event is associated with event based programming and is usually related to an ‘event’ that can happen inside the program. The most common types of events everyone knows is mouse clicking and mouse movement. Event based programming allows for programs to feel more ‘alive’ and allow a program to be interactive with a user and feel less static.

**JIT 2**

Will we be programming events for our projects using C# or are they not done with C# and instead are done via .NET or some other programming language?