1.1 Getting Started: Classes, Types, and Objects

Building data structures and algorithms requires that we communicate detailed instructions to a computer, and an excellent way to perform such communication is using a high-level computer language, such as Java. In this chapter, we give a brief overview of the Java programming language, assuming the reader is somewhat familiar with an existing high-level language, and we continue this discussion in the next chapter, focusing on object-oriented design principles. This book does not provide a complete description of the Java language, however. There are major aspects of the language that are not directly relevant to data structure design, which are not included here. We begin our Java primer with a program that prints “Hello Universe!” on the screen, which is shown in a dissected form in Figure 1.1.

```java
public class Universe {
    public static void main(String[] args) {
        System.out.println("Hello Universe!");
    }
}
```

The main “actors” in a Java program are objects. Objects store data and provide methods for accessing and modifying this data. Every object is an instance of a class, which defines the type of the object, as well as the kinds of operations that it performs. The critical members of a class in Java are the following (classes can also contain inner class definitions, but let us defer discussing this concept for now):

- Data of Java objects are stored in instance variables (also called fields).
  Therefore, if an object from some class is to store data, its class must specify instance variables to do the storage. Instance variables can either come from base types (such as integers, floating-point numbers, or Booleans) or they can refer to objects of other classes.
- The operations that can act on data, expressing the “messages” objects respond to, are called methods. These consist of constructors, procedures, and functions. They define the behavior of objects from that class.

Figure 1.1: A “Hello Universe!” program.