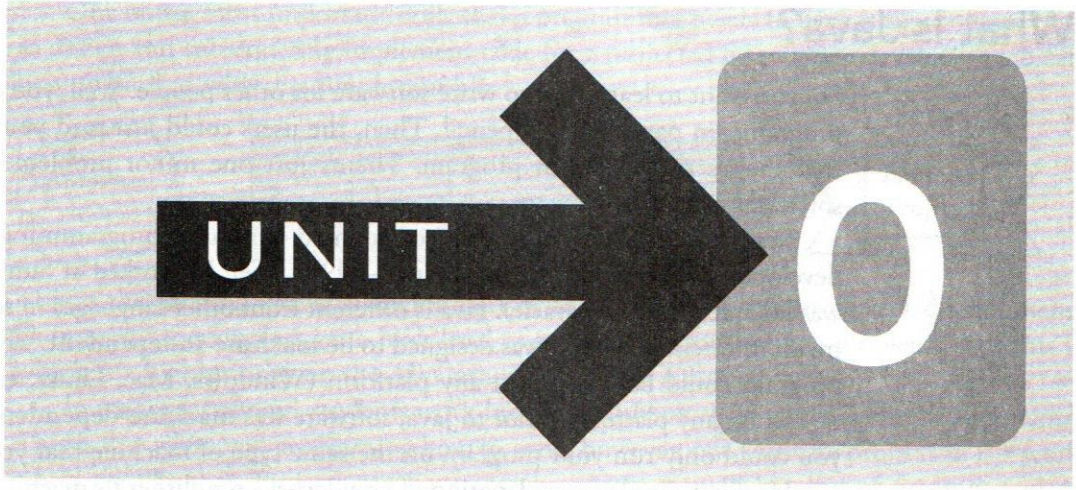


Background on Software Development

IN THIS UNIT

Summary: This unit will give you background on Java and what a software developer does. You will also learn the purpose of software, and the tools you will use to prepare for the exam.

If you take a bird's eye view of the idea of software development, you can see that it is important for programmers to follow guidelines. The rules that guide programmers help make sure that programs are easily maintainable.

Key Ideas

KEY IDEA

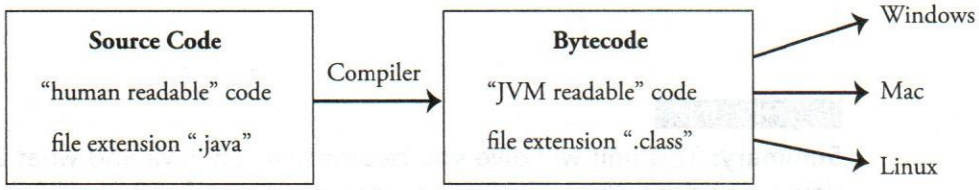
- Java is the programming language of the AP Computer Science A Exam.
- Java is an object-oriented programming language.
- Software developers design and implement code for countless applications.
- An integrated development environment (IDE) is the tool that you will use when you write your Java code.
- Program specifications define what a program is supposed to accomplish.
- Someone other than the software developer often writes program specifications.
- Software engineering is a field that studies how software is designed, developed, and maintained.
- Top-down and bottom-up are two approaches to designing class hierarchy.
- Procedural abstraction helps make software more modular.
- Software developers use testing techniques to verify that their program is working correctly.

What Is Java?

So, you want to learn how to write software for other people. Well, you could write all your programs on paper using a pencil. Then, the users could just read your code and pretend that they're running your program. There's just one minor problem with that: to write software, you need software.

Java is a general-purpose, object-oriented computer programming language that was developed by James Gosling in the mid-1990s when he worked at Sun Microsystems (Sun was later acquired by Oracle). Java is different from other languages in several ways, but the major difference is that it was designed to be **machine-independent**. This means that a Java program could be written on any platform (Windows, Mac, Linux, etc.) and then could be run on any platform. Prior to Java, software was **machine-dependent** which meant that you could only run your program on the same type of machine that you wrote it on.

Here is a simplistic explanation of how Java accomplishes its machine-independent process. Suppose you write a program in Java. The computer can't run the program as you've written it, because it doesn't understand Java. It can only run a program whose instructions are written in **machine code**. A **compiler** is a program that converts your Java program into code called **bytecode**. The compiler checks your program for **syntax errors**, and when all of these errors are eliminated, the compiler generates a new program that is readable by a **Java virtual machine (JVM)**. The JVM is where the Java program actually runs since it translates the bytecode into machine code. The JVM is a part of the **Java Runtime Environment (JRE)** that is provided by Oracle. Oracle writes a JVM for numerous platforms like Windows, which is how a Java program can run on any machine.



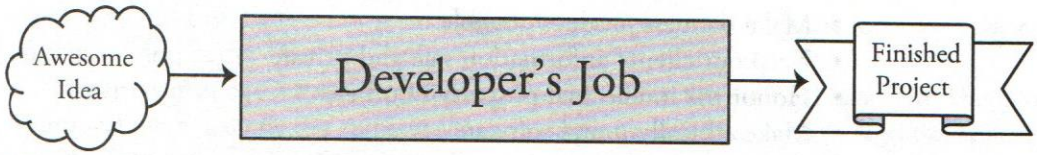
It's a Big Java World Out There
Java has thousands of built-in classes that all Java programmers can use. Since the focus of this book is to help you score a 5 on the AP Computer Science A Exam, only the features that are tested on the exam are included in this book.

What Is a Software Developer?

Software developers are the wizards of our technology-based world. They are the people who design and create the programs that run on smartphones, tablets, smart TVs, computers, cars, and so on. It is stunning how many people the software developer affects.

Software developers may work individually, but most of the time, they work as part of a team of developers. It is essential that they write clean, readable code so that others can read and modify it, if needed. Since other developers will often read your code, it is important to insert comments into your code. In the real world of programming, most programmers despise commenting their code, but it is a necessary evil, and every developer appreciates reading code that has been well documented.

One of the most important skills that a programmer needs to have is figuring out how to design and subsequently implement code for a project. The software developer's job can be summarized with this simple graphic:



This is the exciting, yet challenging part of the developer's role in project development. Computer programmers get a high level of satisfaction from solving problems, and this is why it is a high priority on the AP Computer Science A Exam.

In many classrooms, students work independently on their programming assignments. This is fine when you are learning how to code; however, few software projects in the work world are like that. Projects can have anywhere from a small handful of programmers to hundreds of programmers. There is a hierarchy among the developers that separates the types of work to be done.

What Is OOP (Object-Oriented Programming)?

Object-oriented programming is an approach to programming that uses the concept of classes and objects. It's a brilliant approach since *we live in an object-oriented world.* Everywhere you look, there are objects (cars, balls, trees, etc). These objects have attributes that make them different from other objects (red 2-door convertible, black 4-door hatchback, youth soccer ball, regulation football). Many of these objects can perform some kind of action, or you can get information from them. In OOP, classes define how an object will be constructed. Then, objects are created from these classes, and you manipulate them in your program. Kind of makes you feel like you rule the world, doesn't it?

Viewing the World Through the Eyes of a Software Developer

Something I challenge all my computer science students to do is to begin viewing the world through the eyes of a software developer. As you learn new concepts, try to relate them to things in your daily life. If you are a gamer, start to analyze the games that you play and see if you can figure out how the developers made the game. My students tell me that their minds "open up" after doing this, and it helps them to be better programmers since they are constantly making connections between the computer science topics and the world that they live in.

For the Good of All Humankind

Let's be honest; in our world, there are good software developers and there are evil software developers. The hackers who live on the dark side will always exist, and our job as noble software developers is to make the world a better place. Integrity is vital to our side and is a

badge to be worn with pride. Therefore, we must vow to only write software that is for the good of all humankind and stand up against all evil software developers.

As developers, we strive to:

- Make the best product possible
- Keep our clients' information safe and private
- Honor the intellectual property of others
- Make ethically moral software

Choosing Your IDE

An Integrated Development Environment, or IDE, is a piece of software that allows you to write software more easily. Professional IDEs, such as Eclipse or IntelliJ, have really awesome features that make creating programs easier. Other IDEs are great for introducing a newbie to programming.

There are many excellent IDEs for Java. Your computer science teacher will probably choose the IDE for you. While it is important to learn how to write a program with an IDE, it is also important to remember that *you will not be able to use a computer* on the AP Computer Science A Exam. So, whatever program you use to write your Java code, you *must* learn how to do it without the use of a computer.

No Computer!
Computers are *not used* on the AP Computer Science A Exam.

HelloWorld

In whatever IDE you choose to write your programs, type in the following code for the HelloWorld program. Compile it and run it. This will get you to the point where you can begin this book.

PROGRAM

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

OUTPUT (This is what is displayed on the console screen.)

```
Hello World.
```

Spacing and Indenting in Your Program

Programmers have the freedom to put spacing and indentions in their program in any way they want. Typically, programmers follow spacing guidelines to make it easier to read. The `HelloWorld` example shown above demonstrates a classic form for spacing and indenting code.

You may also see a `HelloWord` program that looks like the next example. This program will run in exactly the same way as the previous example even though the spacing is different.

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

The Software Development Cycle

Program Specifications

When you are in computer science class, your teacher probably gives you assignments that have **program specifications**. This means that in order to receive full credit for the assignment, you have to follow the directions and make sure that your program does what it is supposed to do. When you write your own programs for your own purpose, you make your own specifications.

In the real world of programming, it is common for someone from a design team to give the software developers the program specifications. Specifications are descriptions of what the program should look like and how it should operate. Someone other than the actual programmer writes these **specs**. You may have heard that it is important to be a good communicator. Well, great programmers have good communication skills so that they can talk back and forth with the person who writes the specs to clarify any questions they may have.

Working for Others

Programmers must learn how to write code from specifications. On the Free-response questions section of the AP Computer Science A Exam, you have to write code that matches the specifications that are described in the questions.

Software Engineering

Software engineering is the study of designing, developing, and maintaining software.

Over the years, many different software development models have been designed that help developers create quality software. Here is a brief list.

- Prototyping—an approximation of a final system is built, tested, and reworked until it is acceptable. The complete system is then developed from this prototype.
- Incremental development—The software is designed, implemented, and tested a little bit at a time until the product is finished.
- Rapid application development—the user is actively involved in the evaluation of the product and modifications are made immediately as problems are found. Radical changes in the system are likely at any moment in the process.

- Agile software development—the developers offer frequent releases of the software to the customer and new requirements are generated by the users. Short development cycles are common in this type of development.
- Waterfall model—the progress of development is sequential and flows downward like a waterfall. Steps include conception, initiation, analysis, design, construction, testing, implementation, and maintenance.

Fun Fact: *Margaret Hamilton, a computer scientist who helped develop the on-board flight software for the Apollo space program, coined the phrase "software engineering." Her code prevented an abort of a moon landing!*

So What's the Best Way?

It depends on the situation. It varies from problem to problem, situation to situation, and even company to company. My belief is that you should view the models of software engineering as tools in a toolbox in which the technique that you apply depends on the project you are working on.

Designing Class Hierarchy

Top-Down Versus Bottom-Up Design

Top-down and **bottom-up** are two ways of approaching class hierarchy design. Top-down is also referred to as **functional decomposition**. The two designs are different only in their approach to the problem. Top-down design starts with the big picture, whereas a bottom-up design starts with the details. They each work toward the other, but where they begin is different.

Top-Down Versus Bottom-Up in Object-Oriented Programming

This concept can be applied to designing a complex class hierarchy. If you start your design by thinking of what would be at the top of the class hierarchy, then you are doing a top-down design. If you start your design by thinking of what would be at the bottom of the class hierarchy, then you are doing a bottom-up design.

For example, suppose you have been given the opportunity to design a video game for the NFL (National Football League). If you approach the design from a bottom-up perspective, you would say, "Let's design the Player class first. We'll determine all the instance variables and methods for the Player. Then we'll move on to the Team class. After we get those done, we'll figure out where to go from there." This approach starts with the lowest level object that could be in your game first (the Player), and then works *up* from there.

If you approach the design from a top-down perspective, you would say, "Let's design the League class first. We'll identify what every League has and what it can do. Then, we'll move onto the Conference class. After we get those done, we'll figure out where to go from there." This approach starts with the highest possible object first (the League), and then works *down* from there.

Top-Down Versus Bottom-Up
Top-down design begins with the parent classes, while bottom-up starts with the child classes. On the AP Computer Science A Exam, you will need to be able to recognize a top-down versus a bottom-up design.

Procedural Abstraction

When a program is broken down into pieces and each piece has a responsibility, then the program is following **procedural abstraction**. For example, if you write a quest game and have separate methods for hunting, gathering, and collecting, then you are following procedural abstraction. If your hunt method *also* gathers, collects, and slays dragons while eating a peanut butter and jelly sandwich, then you are not following procedural abstraction.

Fun Fact: *The word* procedure *comes from early procedural programming languages like Fortran and Pascal. Procedures are similar to methods in Java. Each procedure has a purpose and does its job when you ask it to.*

Procedural Abstraction
Every method in Java should have a specific job. Complex tasks should be broken down into smaller tasks.

Testing

You may have heard of video game testers whose job it is to find bugs in the software. Talk about a dream job, right? Well, professional programmers test their work to make sure that it does what it's supposed to do before they release it. There are a few standard ways that developers test code.

Unit Testing

Unit tests are typically short methods that test whether or not a specific method is performing its task correctly. Suppose you have a method that performs some kind of mathematical calculation. To test if this method is working correctly, you would write another method, called a **unit test**. The unit test calls the method you want to test to see if it is producing correct answers. To accomplish this, you figure out what the answer is *before* you make the call to the method and compare the method's answer with your answer.

Here are the steps to creating a unit test for a method:

1. Create a method that performs some kind of calculation.
2. Think of an appropriate input for the method and calculate the answer.
3. Create a unit test. The unit test must contain a call to the original method.
4. Run the unit test with your input and compare the result to the answer you calculated.
5. Repeat this process for every possible type of input that you can think of. For example, when creating a unit test method that does a mathematical calculation, you may want to test it with a positive number, a negative number, and zero.
6. If the method produces correct values for every one of your inputs, the method *passes the unit test.*

Example

Create a unit test method for the `getArea` method of the Circle class. Pass it a value for the radius as well as the known answer for its area to determine if the method is doing its calculation correctly.

```
public class UnitTestForCircle
{
   public static void main(String[] args)
   {
      testGetArea(new Circle(10), 314.15926); // Supply the correct answers
      testGetArea(new Circle(0), 0.0);
   }

   public static void testGetArea(Circle c, double correctAnswer)
   {
      // Use a tolerance to determine closeness of answer
      double tolerance = 0.0001;

      if (Math.abs(c.getArea() - correctAnswer) <= tolerance)
      {
         System.out.println(c.getRadius() + " passed the test");
      }
      else
      {
         System.out.println(c.getRadius() + " failed the test");
      }
   }
}  // End of Circle class
```

```
public class Circle
{
   /*  Additional implementation not shown */

   public double getArea()
   {
      return Math.PI * Math.pow(radius, 2);
   }
}  // End of Circle class
```

```
OUTPUT
10.0 passed the test
0.0 passed the test
```

Integration Testing

Integration testing is used to make sure that connections to outside resources are working correctly. For example, if you are writing software that connects to Snapchat, then you would write a method that tests whether the connection is made correctly.

› Rapid Review

- Java is a general-purpose, object-oriented computer programming language.
- Java source code is translated into bytecode by the compiler.
- Java has thousands of built-in classes that all Java programmers can use.
- A software developer is a person who writes computer programs.
- Software developers must be creative but also pay attention to detail.
- An object-oriented language is a programming model that uses classes and objects.
- To be a better programmer, you should start to view the world through the eyes of a software developer.
- Software developers must keep their clients' information safe, honor the intellectual properties of others, and make ethically moral software.
- An Integrated Development Environment (IDE) is a piece of software that allows you to write software more easily.
- A computer is not used on the AP Computer Science A Exam.
- HelloWorld is traditionally the first computer program that a beginning Java programmer writes.
- Program specifications define what a program is supposed to accomplish.
- Someone other than the software developer often writes program specifications.
- A good software developer has the ability to communicate with others.
- Software engineering is a field that studies how software is designed, developed, and maintained.
- There isn't one correct way to design a computer program. How you write the program depends on the application.
- Top-down and bottom-up are two approaches to designing class hierarchy.
- Top-down design starts with the parent classes and works toward the child classes.
- Bottom-up design starts with the child classes and works toward the parent classes.
- Procedural abstraction helps make software more modular.
- When applying procedural abstraction to a complex task, the developer breaks the complex task down into smaller parts and implements these tasks individually.
- Software developers use testing techniques to verify that their program is working correctly.
- Unit testing is the process of testing a method to make sure that it is working correctly.
- Integration testing is the process of testing connections to outside resources to make sure that they are working correctly.

› Review Questions

1. You are designing a program to simulate the students at your school. Starting with which one of these classes would best demonstrate top-down design?

(A) The student body
(B) The senior class
(C) The girls in the senior class
(D) The girls taking programming in the senior class
(E) Mary Zhang, a girl in the programming class

2. If you were designing a unit test for the `isEven` method, which of these methods would you prefer to be working with and why? Note that the two methods both work and return the same result.

Option 1:

```
public boolean isEven(int num)
{
   int digit = num % 10;
   if (digit >= 5)
      if (digit == 6 || digit == 8)
         return true;
      else
         return false;
   else if (digit == 0 || digit == 2 || digit == 4)
      return true;
   else
      return false;
}
```

Option 2:

```
public boolean isEven(int num)
{
   return (num % 2 == 0);
}
```

(A) Option 1 because it doesn't use modulus, so it's easier to understand.
(B) Option 1 because it is longer.
(C) Option 2 because there is only one path through the code, so fewer test cases are needed.
(D) Neither option requires testing because the intent of the method is clear from the name.
(E) It doesn't matter. Testing them would be the same.

3. What is the term used to describe breaking a large program into smaller, well-defined sections that are easier to code and maintain?

(A) Encapsulation
(B) Procedural abstraction
(C) Inheritance
(D) Static design
(E) Polymorphism