COMP112/18 - Programming I

02 Using Java

Instructor: Ke Wei（柯韋）

A319 ✆ Ext. 6452 ✉ wke@ipm.edu.mo

http://brouwer.ipm.edu.mo/COMP112/18/

Bachelor of Science in Computing, School of Public Administration, Macao Polytechnic Institute

August 31, 2018

Outline

A Java Program

```
1  import java.util.Scanner; // refer to a class (module) in some other package
2  public class SquareRoot { // the main class for this program
3      public static void main(String[] args) { // the main method
4          double x; // variable declaration
5          try ( Scanner scanner = new Scanner(System.in) ) {
6          // create an object (scanner) for calling system input service
7              x = scanner.nextDouble(); // input x as a real number
8          } // close and discard the scanner
9          if ( x >= 0.0 ) { // start a conditional statement
10             double y = Math.sqrt(x); // compute the square root and store the result
11             System.out.println("The square root of "+x+" is "+y); // output the result
12         } else // output the error message
13             System.out.println("Error: the input is a negative numnber");
14     } // end of the main method
15 } // end of the main class
```

Elements of a Java Program

- A Java program consists of a number of *classes*.
- Each class has a class name, and defines some variables and *methods*.
- One of the classes contains a unique *static* method with name "`main`". This method is called the main method of the program. A program starts from its main method. The class containing the main method is sometimes called the main class.
- In a method, there are *statements*, which include variable declarations, assignments, *invocations* to methods and control flow statements. This is where you write your instructions.
- Several related classes can be grouped in a package. Two classes can have the same class name if they are contained in different packages.
- If a class is in a package, it must be referred to with the package name as a *qualifier*, "`packagename.ClassName`". A class can be referred to by only its class name if it is *imported* from its package first.

Classes and Static Methods

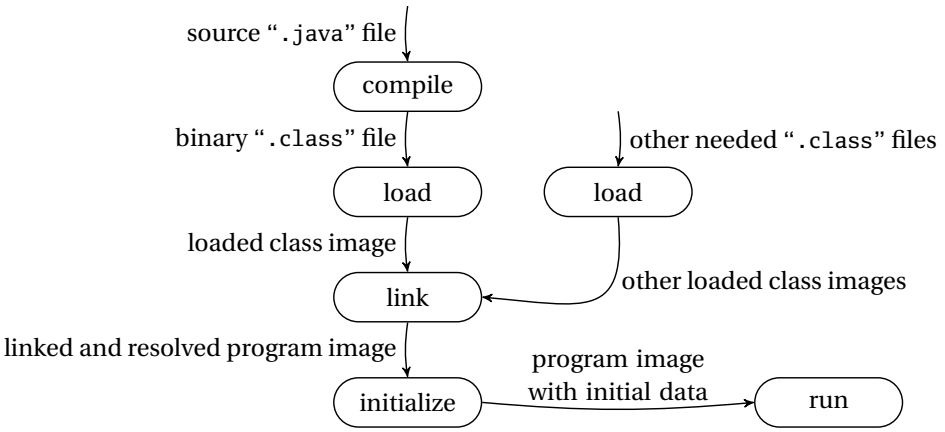
- A class in a Java program serves as a module.
- A class can define a number of variables and methods.
- Static variables and methods can be referred to by their names within the same class. The class name must be included as a *qualifier* if the static variable or static method is from a different class.
- We use only static variables and methods at this stage of learning.
- If we write our code only in the main method, we declare variables within the method. These variables are local to the method.
- Static variables are useful when we have several static methods that need to share those variables.

Java Source Files

- Source files store our programs written in high-level programming languages.
- A source file is usually a text file, with the binary numbers in the file representing characters, and the characters are arranged line by line.
- A text file can be opened, saved and displayed by a *text editor*, such as the *Notepad* comes with Windows. Notepad++ is a much better free general text editor.
- A Java source file contains only one top level class definition, together with some declarations for imported classes.
- The filename of a Java source file ends with "`.java`".

Java Compilation and Execution Process

This diagram shows how a Java program is processed from the source form.



Compiling and Running a Java Program using CLI

To compile and run a program using the command line interface, we follow these steps:

1. Create your Java main class source file using a text editor, and save it to a file with extension "`.java`".
2. Change to the folder (using command "`cd`") containing the source file.
3. Locate your JDK folder path, for example, "`C:\Program Files\Java\jdk-10.0.2`".
4. Type the command to compile:

   your-jdk-path`\bin\javac YourClassName.java`
5. If your program compiles with no errors, verify there's a binary "`.class`" file "`YourClassName.class`" in the same folder as your source file.
6. Type the command to run:

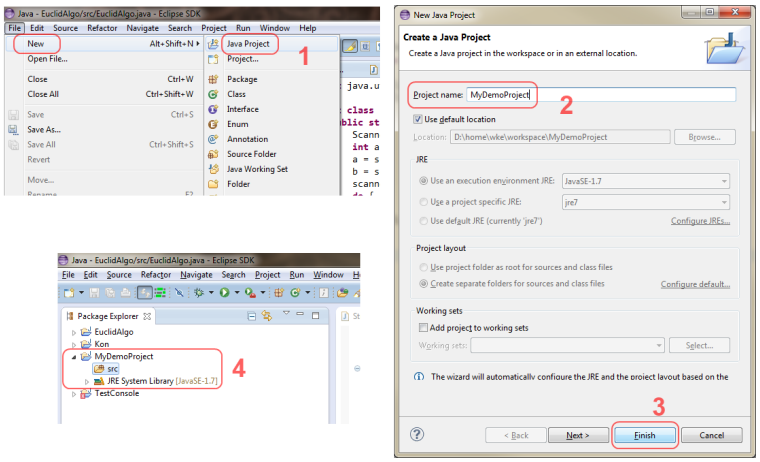
   ```
   java YourClassName
   ```


Developing a Java Program using Eclipse

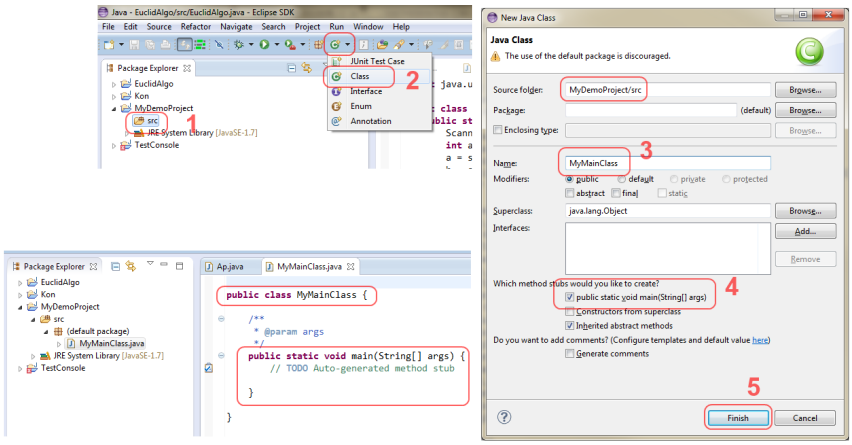
To develop a simple single method Java program using Eclipse, we follow these steps:

1. Create a Java project for your program. A Java project resides in a folder with the project name as the folder name, for example, "`MyProject`". In the project folder, there are two subfolders, "`src`" and "`bin`", storing source "`.java`" files and binary "`.class`" files, respectively.
2. Create the main class with the main method of your program. The main method is where your program starts. The main class source is saved to the source "`.java`" file of the class.
3. Write your program in the main class source file. This includes importing necessary classes before the main class, and writing statements in the main method.
4. Run and test your program. Once your main class source file does not have any syntax errors, you can run the program, supply needed input, and see the result.

Creating a Java Project

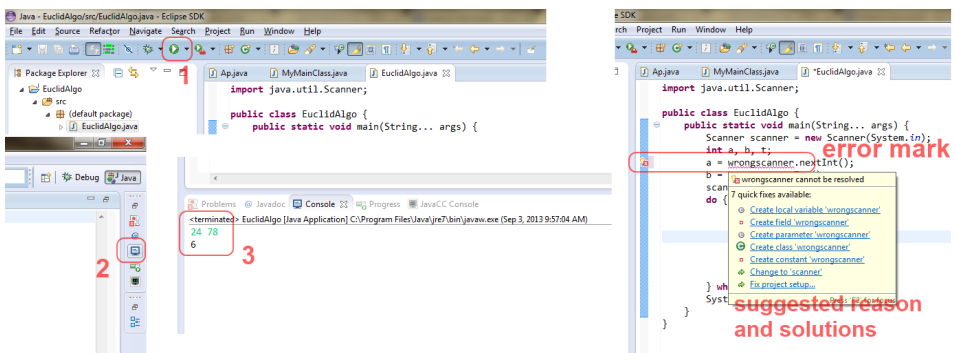


Adding a Class with the Main Method

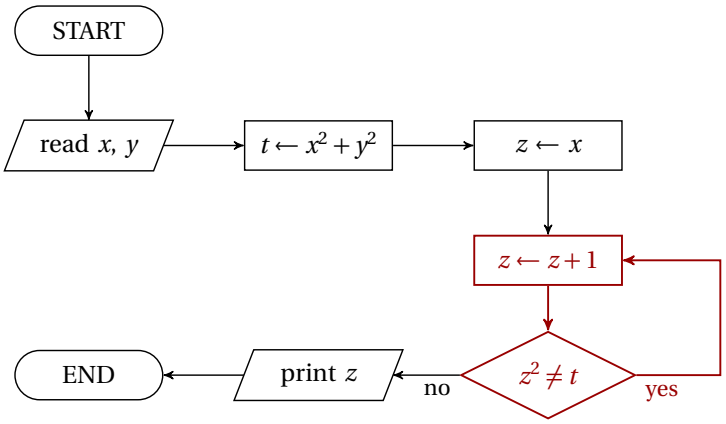


Interacting with the Console Window

When you save your program source file, the program is compiled automatically by Eclipse, and the binary "`.class`" file is put in the "`bin`" folder. If the program has no errors (no red marks), we can run the program and supply input and see output from the console window.



Finding Pythagorean Triples — Flow



Finding Pythagorean Triples — Java Code

```
1  import java.util.Scanner;
2  public class Pythagorean {
3      public static void main(String[] args) {
4          int x, y;
5          try ( Scanner scanner = new Scanner(System.in) ) {
6              x = scanner.nextInt();
7              y = scanner.nextInt();
8          }
9          int t = x*x+y*y, z = x;
10         do {
11             z = z+1;
12         } while ( z*z != t );
13         System.out.println(z);
14     }
15 }
```


Reading Homework

Textbook

- Section 1.6 – 1.8, 1.10, 1.12.

Internet

- Programming language (`http://en.wikipedia.org/wiki/Programming_language`).
- Java (`http://en.wikipedia.org/wiki/Java_(programming_language)`).

Self-test

1.22 – 1.35 (`http://tiger.armstrong.edu/selftest/selftest9e?chapter=1`).