# Appendix

# javadoc

- Javadoc
  - Appendix H (fore reference)
- Demo
  - javadoc *.java
  - javadoc -tag author *.java
  - Javadoc -tag author -tag precondition *.java

# How to generate specifications

**search**

public static boolean **search**(double[] data, double target)

**Author:**

Huiping Cao

**Parameters:**

data - The array that need to be searched

target - The number that need to be searched

**Precondition**

data array must have at least one element

**Postcondition:**

check whether target exists in data or not

**Returns:**

A boolean value denotes whether target exists in the array or not

**Throws:**

java.lang.IllegalArgumentException - indicates whether data is valid

# Document comments

```
/**
    * @author Huiping Cao
    * @param data
    *    The array that need to be searched
    * @param target
    *    The number that need to be searched
    * @precondition
    *    data array must have at least one element
    * @postcondition
    *    check whether target exists in data or not
    * @return
    *    A boolean value denotes whether target exists in the array or not
    * @throws
    *    IllegalArgumentException indicates whether data is valid
    */
```

# javadoc

- Javadoc: Documentation tool to process documentation comments. Javadoc is distributed with JDK
- Javadoc works
  - Read document comments from java files
  - Produce document pages in hypertext markup language (html) format

```
javadoc –tag author -tag param
      -tag precondition:a:"Precondition"
      -tag postcondition:a:"Postcondition:"
      -tag return
      -tag throws
      egjavadoc.java
```

# Java: Try-catch-exception

- Try, catch, throw Appendix C
- An exception is a Java Object that is to indicate abnormal conditions.
- Throwing an exception: a process: when a method detects an abnormal condition
  - it creates an exception object and
  - pass this object upward to the place where the method was activated.

# Throw an exception

- The general form of throwing an exception uses the keyword throw, following this pattern:

throw new *the type of the exception(“an error message”);*

# Catching an Exception

- When an exception occurs and the programmer has some way to handle the problem. In this case, the programmer can use try-catch blocks of the following form:
- All the codes in the try block are executed

```
try{
        //statements that might cause an exception to be thrown
}
catch(Type of the possible exception e){
        //statements to handle the problem.
        //e is the name of the exception object.
        //e.g., e.toString() is a message that is attached to the exception.
}
```

# Use assertions

- Assertions can be turned on or off

Enable the –enableassertions option (or –ea) for the Java runtime system.

```
java –ea <javaprogram>
```

- Do not use assertions to check precondition
  - Reason: assertions can be turned on and off