Practical no. 7

FS19CO042

Aim: Generate complete Javadocs for any two of the above experiments

Tool used: Editor (Notepad/Intellij IDE), JDK and JRE

Theory:

Javadoc

Javadoc is a tool which comes with JDK and it is used for generating Java code documentation in HTML format from Java source code, which requires documentation in a predefined format.

Following is a simple example where the lines inside /*….*/ are Java multi-line comments. Similarly, the line which preceeds // is Java single-line comment.

Example

```
/**
* The HelloWorld program implements an application that
* simply displays "Hello World!" to the standard output.
*
* @author  Omkar Phansopkar
* @version 1.0
* @since   2014-03-31
*/
public class HelloWorld {
   public static void main(String[] args) {
      // Prints Hello, World! on standard output.
      System.out.println("Hello World!");
   }
}
```

You can include required HTML tags inside the description part. For instance, the following example makes use of <h1>....</h1> for heading and <p> has been used for creating paragraph break −

Example

```
/**
* <h1>Hello, World!</h1>
* The HelloWorld program implements an application that
* simply displays "Hello World!" to the standard output.
* <p>
* Giving proper comments in your program makes it more
* user friendly and it is assumed as a high quality code.
*
*
* @author  Omkar Phansopkar
* @version 1.0
* @since   2014-03-31
```

```
*/
public class HelloWorld {


   public static void main(String[] args) {
      // Prints Hello, World! on standard output.
      System.out.println("Hello World!");
   }
}
```

The javadoc Tags

The javadoc tool recognizes the following tags −

Tag	Description	Syntax
@author	Adds the author of a class.	@author name-text
{@code}	Displays text in code font without interpreting the text as HTML markup or nested javadoc tags.	{@code text}
{@docRoot}	Represents the relative path to the generated document's root directory from any generated page.	{@docRoot}
@deprecated	Adds a comment indicating that this API should no longer be used.	@deprecated deprecatedtext
@exception	Adds a **Throws** subheading to the generated documentation, with the classname and description text.	@exception class-name description
{@inheritDoc}	Inherits a comment from the **nearest** inheritable class or implementable interface.	Inherits a comment from the immediate surperclass.
{@link}	Inserts an in-line link with the visible text label that points to the documentation for the specified package, class, or member name of a referenced class.	{@link package.class#member label}
{@linkplain}	Identical to {@link}, except the link's label is displayed in plain text than code font.	{@linkplain package.class#member label}
@param	Adds a parameter with the specified parameter-name followed by the specified description to	@param parameter-name description

	the "Parameters" section.	
@return	Adds a "Returns" section with the description text.	@return description
@see	Adds a "See Also" heading with a link or text entry that points to reference.	@see reference
@serial	Used in the doc comment for a default serializable field.	@serial field-description | include | exclude
@serialData	Documents the data written by the writeObject( ) or writeExternal( ) methods.	@serialData data-description
@serialField	Documents an ObjectStreamField component.	@serialField field-name field-type field-description
@since	Adds a "Since" heading with the specified since-text to the generated documentation.	@since release
@throws	The @throws and @exception tags are synonyms.	@throws class-name description
{@value}	When {@value} is used in the doc comment of a static field, it displays the value of that constant.	{@value package.class#field}
@version	Adds a "Version" subheading with the specified version-text to the generated docs when the -version option is used.	@version version-text

Command to generate html pages from javadocs:

javadoc -d .\pathToDestination .\pathToSrc.java file.

Code:

File 1, exp7a.java

```
import java.util.Scanner;

/**
 * in this class we are adding the square root of indivisual numbers;
 */
public class exp7a{
   /**
    * in this method we are taking the numbers as input and returning the addtion to main function;
    * @param x;
    * @return to main;
    */
   public static int add(int ...x){
      return x[0]*x[0]+x[1]*x[1];
   }
   /**
    * this is the main method where the calling to add function is done and printing the result;
    * @param args
    */
   public static void main(String args[]){
     //creating Scanner class object and passing system.in ;
    Scanner sc= new Scanner(System.in);
    System.out.println("enter the first number:");
      int n1=sc.nextInt();
      System.out.println("enter the second number:");
      int n2=sc.nextInt();
      int a=exp7a.add(n1,n2);
      System.out.print("the addtion of square root of indivisual is :"+a);
   }
}
```

File 2, exp7b.java

```
import java.util.Arrays;

/**

 * this is a assignment7 class for sorting the array;

 */

public class exp7b{

   /**

    * This is the main method where the arrays sorted and printed;

    *@param args

    */

   public static void main(String args[]) {

       System.out.println("sorting the array.....");

       System.out.println("sorted array:");

       Arrays.sort(args);

       for(String i:args)

          System.out.println(i);

   }
}
```

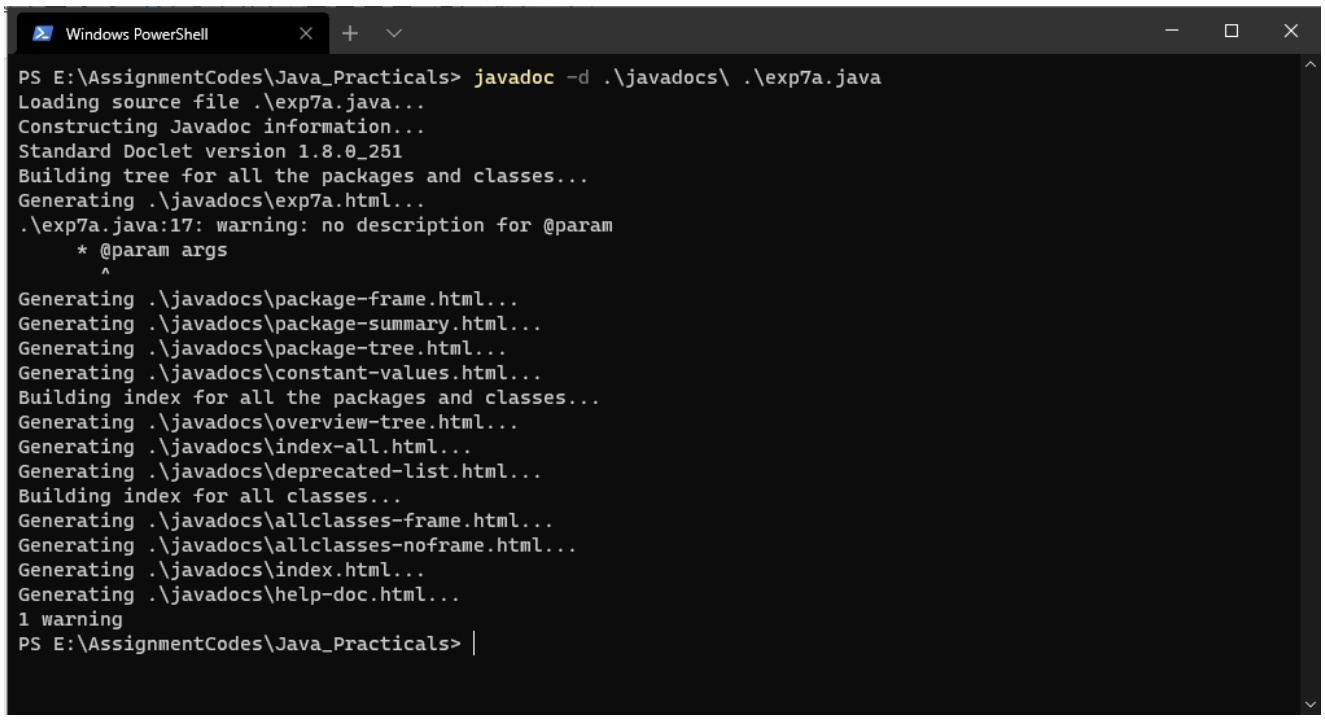
Generating docs:
Perform following commands to generate doc:
javadoc -d .\pathToDestination .\pathToSrc.java file.

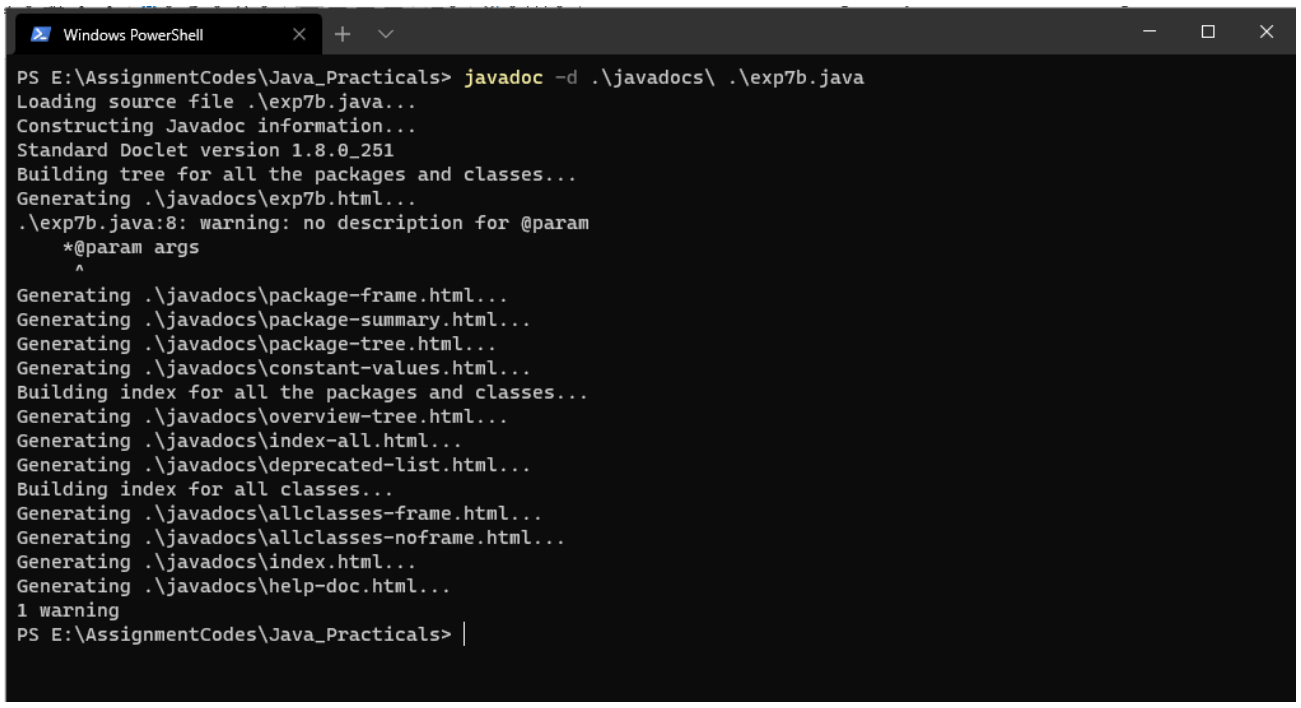
Actual commands:

javadoc -d .\javadocs\ .\exp7a.java

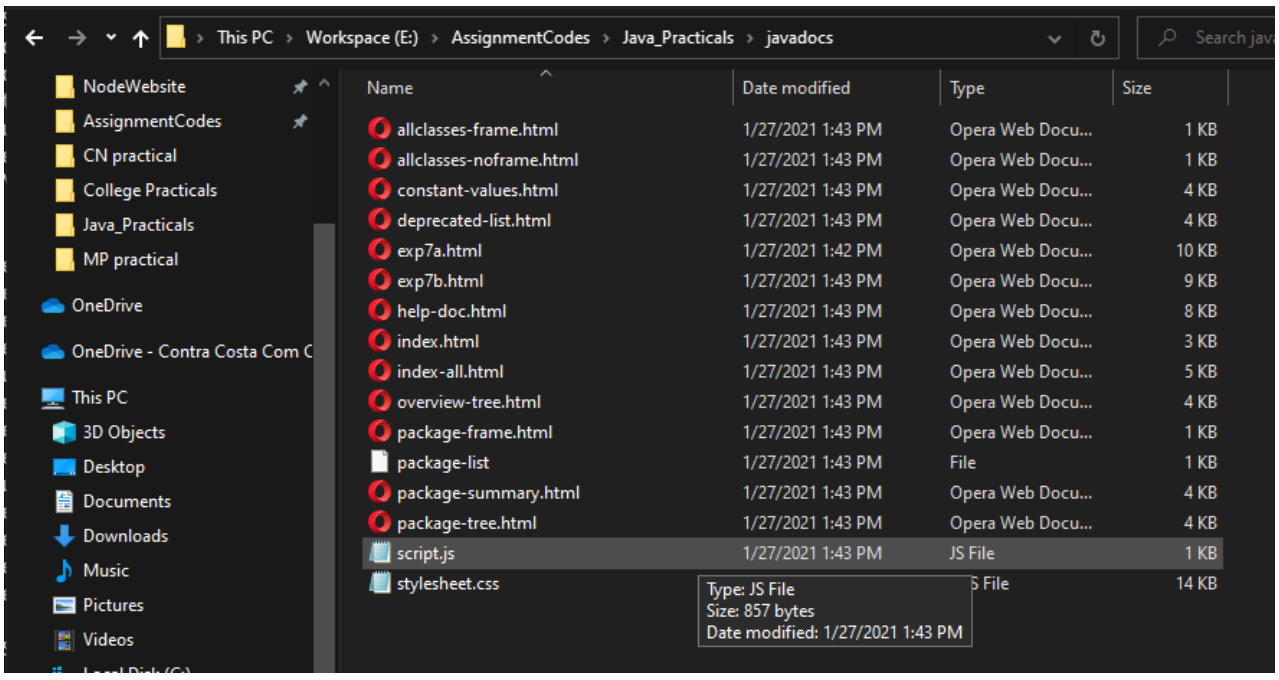
javadoc -d .\javadocs\ .\exp7b.java

Console output:





Docs Output:



file:///E:/AssignmentCodes/Java_Practicals/javadocs/exp7a.html

PACKAGE CLASS TREE DEPRECATED INDEX HELP

PREV CLASS NEXT CLASS FRAMES NO FRAMES ALL CLASSES

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Class exp7a

java.lang.Object
    exp7a

---

public class ***exp7a***
extends java.lang.Object

in this class we are adding the square root of indivisual numbers;

Constructor Summary

Constructors

Constructor and Description
exp7a()

Method Summary

All Methods **Static Methods** **Concrete Methods**

Modifier and Type	Method and Description
static int	***add***(*int... x*) in this method we are taking the numbers as input and returning the addtion to main function;
static void	***main***(*java.lang.String[] args*) this is the main method where the calling to add function is done and printing the result;

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

file:///E:/AssignmentCodes/Java_Practicals/javadocs/exp7b.html

PACKAGE CLASS TREE DEPRECATED INDEX HELP

PREV CLASS NEXT CLASS FRAMES NO FRAMES ALL CLASSES

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Class exp7b

java.lang.Object
    exp7b

---

public class ***exp7b***
extends java.lang.Object

this is a assignment7 class for sorting the array;

Constructor Summary

Constructors

Constructor and Description
exp7b()

Method Summary

All Methods **Static Methods** **Concrete Methods**

Modifier and Type	Method and Description
static void	***main***(*java.lang.String[] args*) This is the main method where the arrays sorted and printed;

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

Conclusion: Thus we understood and successfully created javadocs for our project using various techniques used for commenting and documenting java experiments.