CS 602 Java, HW#1(Individual)
Charitha Kammari(ck369) - Group H

1. Define Java Applet, Java Program, Java Script, and Angular Java Script. Provide examples of each. Name three NYC companies that hire Java, job description, annual salary. (1 point)

Answer:

Definitions:

Java Applet: A Java Applet is a small software created with the Java programming language that is usually embedded into a webpage. It facilitates interactive features and content on webpages by working inside the boundaries of a web browser. However, because of security concerns and improvements in web technology, Java Applets are becoming less common in modern online programming.

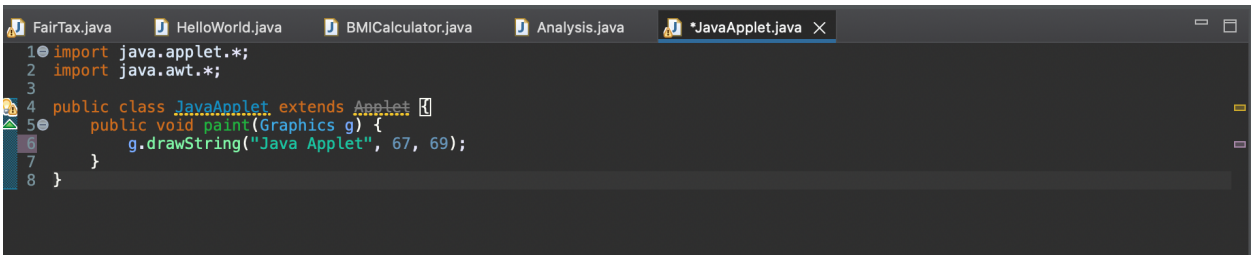
Java Program: A Java program is any software application or program written in the Java programming language that is compiled into bytecode for the Java Virtual Machine (JVM) to run on. Both basic command-line apps and complex enterprise-grade software systems can be developed more easily with Java programming.

JavaScript: JavaScript is a high-level, dynamic programming language that is frequently used to create interactive websites by means of interpretation. JavaScript, in contrast to Java, is mostly used as a client-side scripting language in web development, adding dynamic behavior and interactivity to websites.

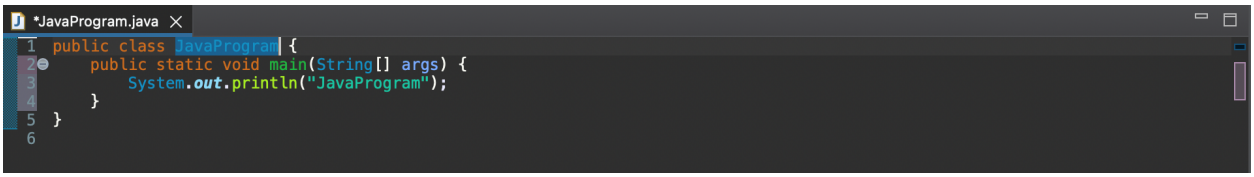
Angular JavaScript: This JavaScript-based open-source front-end web framework, also known as AngularJS, is primarily managed by Google in collaboration with a community of contributors. It is employed in the development of dynamic web applications, the addition of new properties to HTML known as directives, and the use of expressions to link data to HTML elements.

Examples:

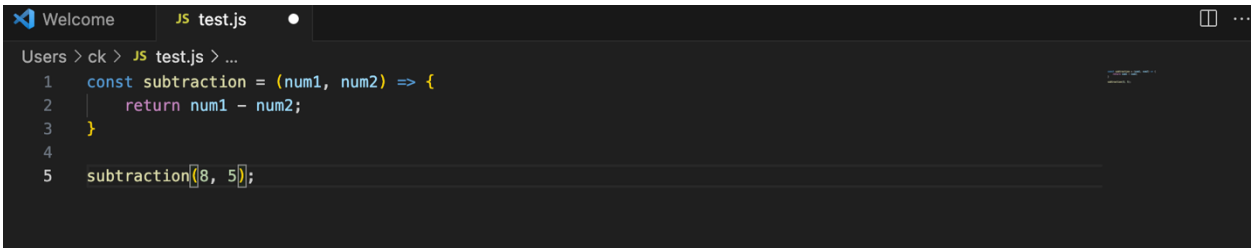
Java Applet:



Java Program:



JavaScript:



Angular JavaScript:

Three NYC companies that hire Java, job description, annual salary:

- Goldman Sachs:
  Job Description: At Goldman Sachs, our Engineers don't just make things – we make things possible. Change the world by connecting people and capital with ideas. Solve the most challenging and pressing engineering problems for our clients. Join our engineering teams that build massively scalable software and systems, architect low latency infrastructure solutions, proactively guard against cyber threats, and leverage machine learning alongside financial engineering to continuously turn data into action. Create new businesses, transform finance, and explore a world of opportunity at the speed of markets.

  Engineering is at the critical center of our business, and our dynamic environment requires innovative strategic thinking and immediate, real solutions. Goldman Sachs Engineers are innovators and problem-solvers, building solutions in risk management, big data, mobile and more. We look for creative collaborators who evolve, adapt to change, and thrive in a fast-paced global environment. Want to push the limit of digital possibilities? Start here.
  Salary Range:
  The expected base salary for this New York, New York, United States-based position is $115000-$180000. In addition, you may be eligible for a discretionary bonus if you are an active employee as of fiscal year-end.

- Bank Of America:
  Job Description:
  Position Summary Bank of America seeks a Java developer to join the Institutional Retirement Technology organization. Responsible for technical design and development of his/her assignments. Experience in all phases of software development lifecycle including analysis, design, implementation, testing and deployment. Take a long-term strategic approach and creation of highly functional products and reusable components. Implement technical solutions that involve diverse development platforms, software, hardware, backend and front-end technologies. Work closely with the manager, team lead, peers, business owners, analysts and QAs to build features that span various parts of the system and retirement products. Collaborate and support the implementation of software applications. Excellent verbal and written communication. Required Skills Bachelor's degree in Computer Science or related field Plus 4 years of IT experience in application design and development, and experience in various technical aspects of the development. Must have prior experience in software design and development for both front-end and backend components utilizing Core Java, AngularJS, Rest API, and Oracle relational database. Strong understanding of multi-tenant software as a service architecture. Excellent verbal and written communication skills. Desired Skills Python and Kafka would be a plus.
  Salary Range:
  $100K -- $150K

- Sumitomo Mitsui Banking Corporation – SMBC Group:
  Job Description:
  SMBC Group is a top-tier global financial group. Headquartered in Tokyo and with a 400-year history, SMBC Group offers a diverse range of financial services, including banking, leasing, securities, credit cards, and consumer finance. The Group has more than 130 offices and 80,000 employees worldwide in nearly 40 countries. Sumitomo Mitsui Financial Group, Inc. (SMFG) is the holding company of SMBC Group, which is one of the three largest banking groups in Japan. SMFG's shares trade on the Tokyo, Nagoya, and New York (NYSE: SMFG) stock exchanges.

  In the Americas, SMBC Group has a presence in the US, Canada, Mexico, Brazil, Chile, Colombia, and Peru. Backed by the capital strength of SMBC Group and the value of its relationships in Asia, the Group offers a range of commercial and investment banking services to its corporate, institutional, and municipal clients. It connects a diverse client base to local markets and the organization's extensive global network. The Group's operating companies in the Americas include Sumitomo Mitsui Banking Corp. (SMBC), SMBC Nikko Securities America, Inc., SMBC Capital Markets, Inc., SMBC Rail Services LLC, Manufacturers Bank, JRI America, Inc., SMBC Leasing and Finance, Inc., Banco Sumitomo Mitsui Brasileiro S.A., and Sumitomo Mitsui Finance and Leasing Co., Ltd.

The anticipated salary range for this role is between $103,000.00 and $206,000.00. The specific salary offered to an applicant will be based on their individual qualifications, experiences, and an analysis of the current compensation paid in their geography and the market for similar roles at the time of hire. The role may also be eligible for an annual discretionary incentive award. In addition to cash compensation, SMBC offers a competitive portfolio of benefits to its employees.
Salary Range:
$103,000/yr - $206,000/yr

2. When you compile Java programs on PC, there are two ways. One is the DOS format, and another one is Eclipse IDE. Download both platforms onto your personal PC. Describe step by step process for downloading. Show screenshots to prove both ways. (2 points)
Answer:

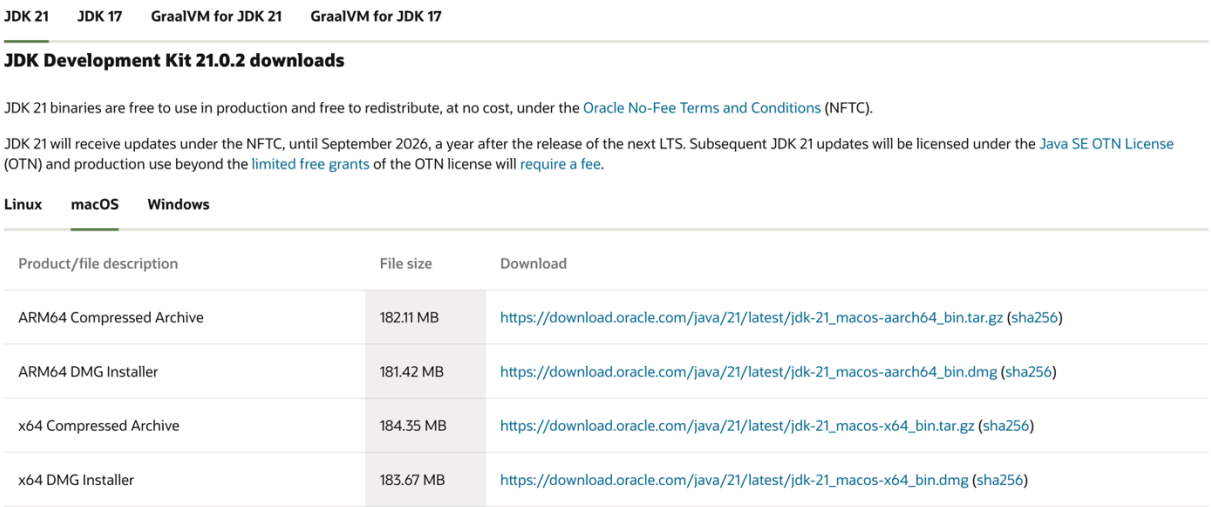
The Java Development Kit (JDK) and the Java Runtime Environment (JRE) must be downloaded in order for Java programs to run on a PC.

1) To run DOS-formatted Java programs:

Use the javac command to begin compiling the Java program.

After that, use the java command and the main class name to run the compiled application.

The steps in this sequence are to use the javac command to compile the Java source code into bytecode, then the java command to run the produced code.



JDK 21 **JDK 17** **GraalVM for JDK 21** **GraalVM for JDK 17**

JDK Development Kit 21.0.2 downloads

JDK 21 binaries are free to use in production and free to redistribute, at no cost, under the Oracle No-Fee Terms and Conditions (NFTC).

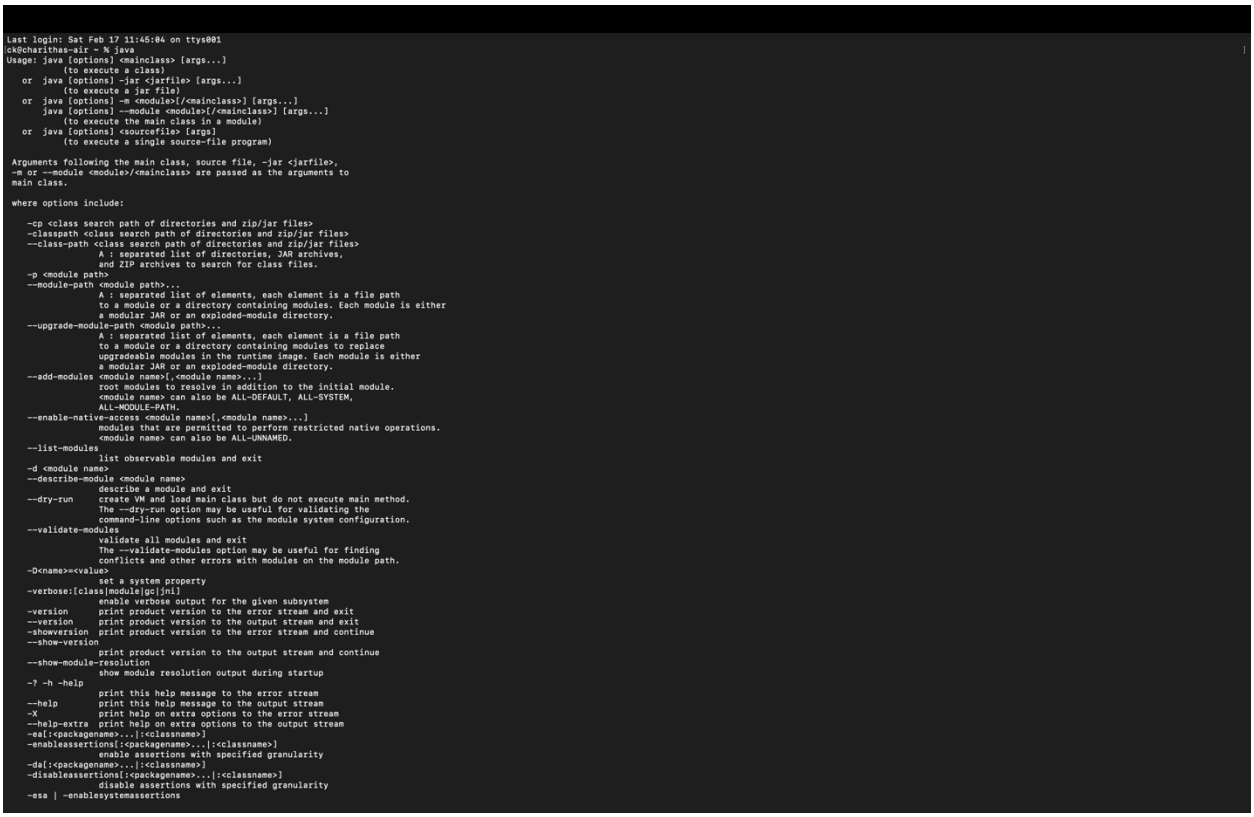
JDK 21 will receive updates under the NFTC, until September 2026, a year after the release of the next LTS. Subsequent JDK 21 updates will be licensed under the Java SE OTN License (OTN) and production use beyond the limited free grants of the OTN license will require a fee.

Linux **macOS** **Windows**

Product/file description	File size	Download
ARM64 Compressed Archive	182.11 MB	https://download.oracle.com/java/21/latest/jdk-21_macos-aarch64_bin.tar.gz (sha256)
ARM64 DMG Installer	181.42 MB	https://download.oracle.com/java/21/latest/jdk-21_macos-aarch64_bin.dmg (sha256)
x64 Compressed Archive	184.35 MB	https://download.oracle.com/java/21/latest/jdk-21_macos-x64_bin.tar.gz (sha256)
x64 DMG Installer	183.67 MB	https://download.oracle.com/java/21/latest/jdk-21_macos-x64_bin.dmg (sha256)

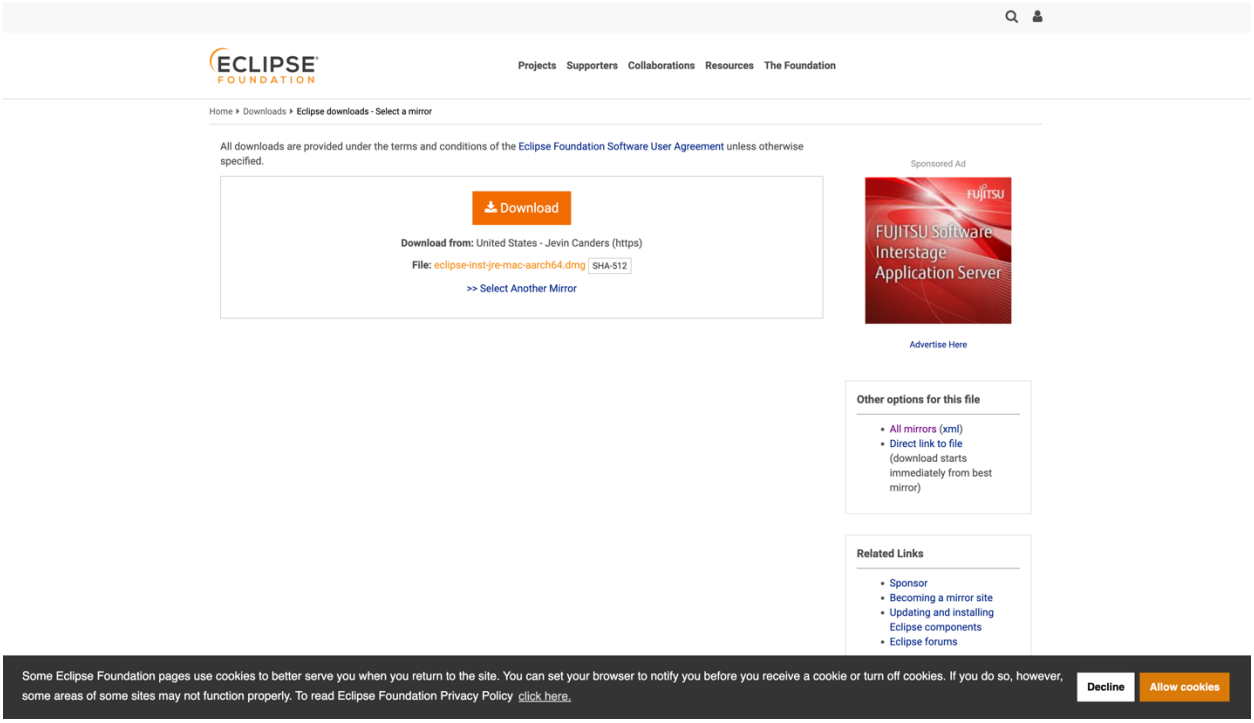

Install JDK 21.0.2
Welcome to the JDK 21.0.2 Installer
Introduction
Destination Select
Installation Type
Installation
Summary
This program will guide you though the installation process for the Java SE Development Kit 21.0.2.
Java
ORACLE
Go Back
Continue

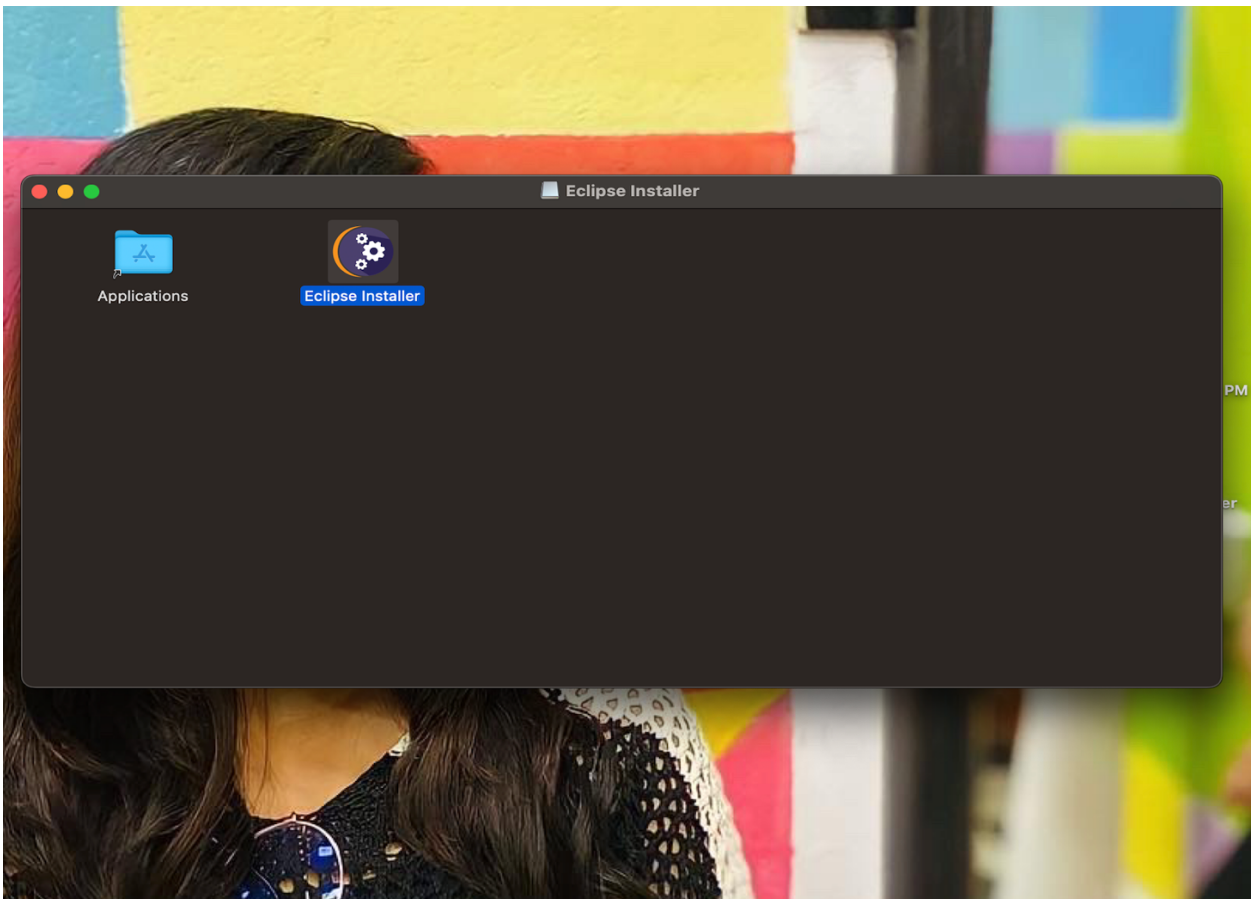

Install JDK 21.0.2
The installation was completed successfully.
Introduction
Destination Select
Installation Type
Installation
Summary
Install Succeeded
Visit the Java SE Documentation to access tutorials, API documentation, developer guides, release notes and more to help you get started with the JDK.
Java
ORACLE
Go Back
Close

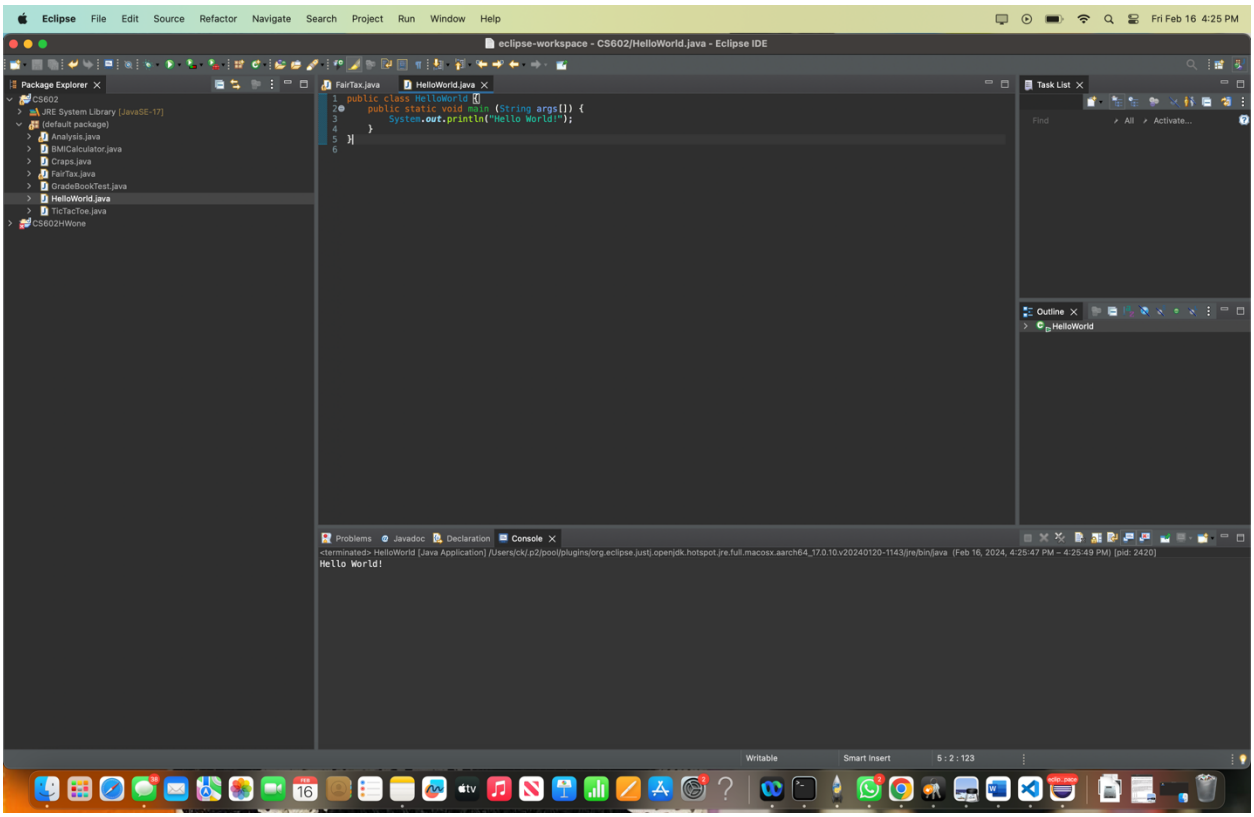


2) To run using eclipse

Open Eclipse first, then set up a workspace to arrange your apps. Next, make a class file in the program's workspace that you are now working on. The software can now be run by selecting the green play button from the toolbar. You can create and test Java programs more quickly with the Eclipse IDE thanks to this simplified procedure.


ECLIPSE FOUNDATION
Projects Supporters Collaborations Resources The Foundation
Home ▸ Downloads ▸ Eclipse downloads - Select a mirror
All downloads are provided under the terms and conditions of the Eclipse Foundation Software User Agreement unless otherwise specified.
Download
Download from: United States - Jevin Canders (https)
File: eclipse-inst-jre-mac-aarch64.dmg SHA-512
>> Select Another Mirror
Sponsored Ad
FUJITSU
FUJITSU Software Interstage Application Server
Advertise Here
Other options for this file
All mirrors (xml)
Direct link to file (download starts immediately from best mirror)
Related Links
Sponsor
Becoming a mirror site
Updating and installing Eclipse components
Eclipse forums
Some Eclipse Foundation pages use cookies to better serve you when you return to the site. You can set your browser to notify you before you receive a cookie or turn off cookies. If you do so, however, some areas of some sites may not function properly. To read Eclipse Foundation Privacy Policy click here.
Decline
Allow cookies


Eclipse Installer
Applications
Eclipse Installer
PM



3. Use Eclipse, create, compile, run, print (program listing and results) and explain:

b. How Analysis.java works, pages 93-94 (1 point)

Answer:

Description:

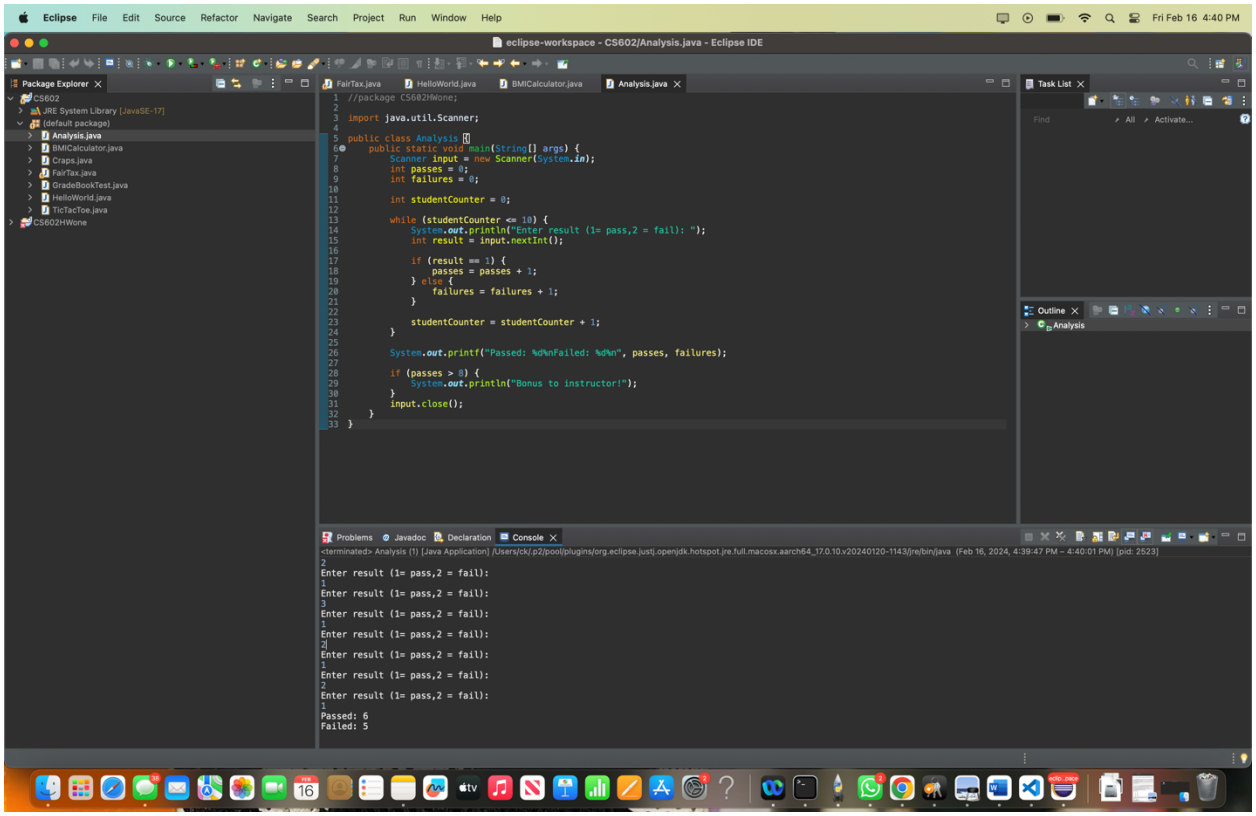
The purpose of this Java software, "Analysis," is to collect and evaluate the results of a theoretical investigation or assessment. This is a brief synopsis of how it works:

The application uses the java.util package's Scanner class to handle user input.
Main Method: To begin, a Scanner object called "input" is initialized in order to record user answers. The integer variables "passes" and "failures" are then initialized to record the number of students that passed and failed, respectively. It also configures "studentCounter" to track the quantity of students handled.
Looping Logic: The application asks the user to enter pass (1) or fail (2) for each student inside of a while loop. The user's reaction is then recorded and captured, and "passes" or "failures" are incremented correspondingly.

After assessing every student, the software shows the overall numbers of passes and fails. The application sends out a "Bonus to instructor!" message to recognize this accomplishment if the pass count is higher than eight.

In summary, this application provides a summary of pass and fail tallies, simulating the grading procedure for ten students. In the event that pass rates are exceptionally high, it also gives the teacher kudos. Its usefulness might be further improved with additions like accurate handling for ten students and more understandable user instructions.

Code:



d. How Craps.java works, pages 171-173 (1 point)

Answer:

Description:

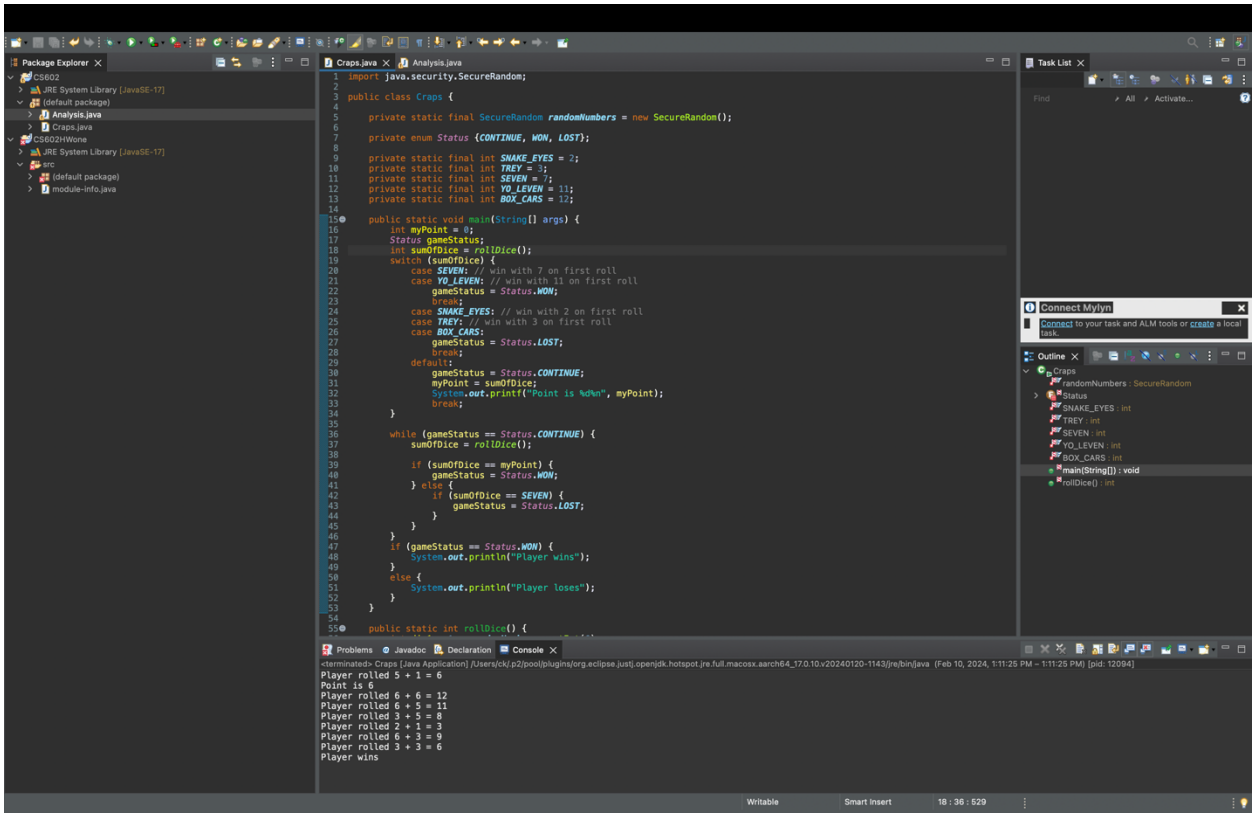
Craps, a Java software, simulates the well-known dice game. This is a brief synopsis of how it functions:

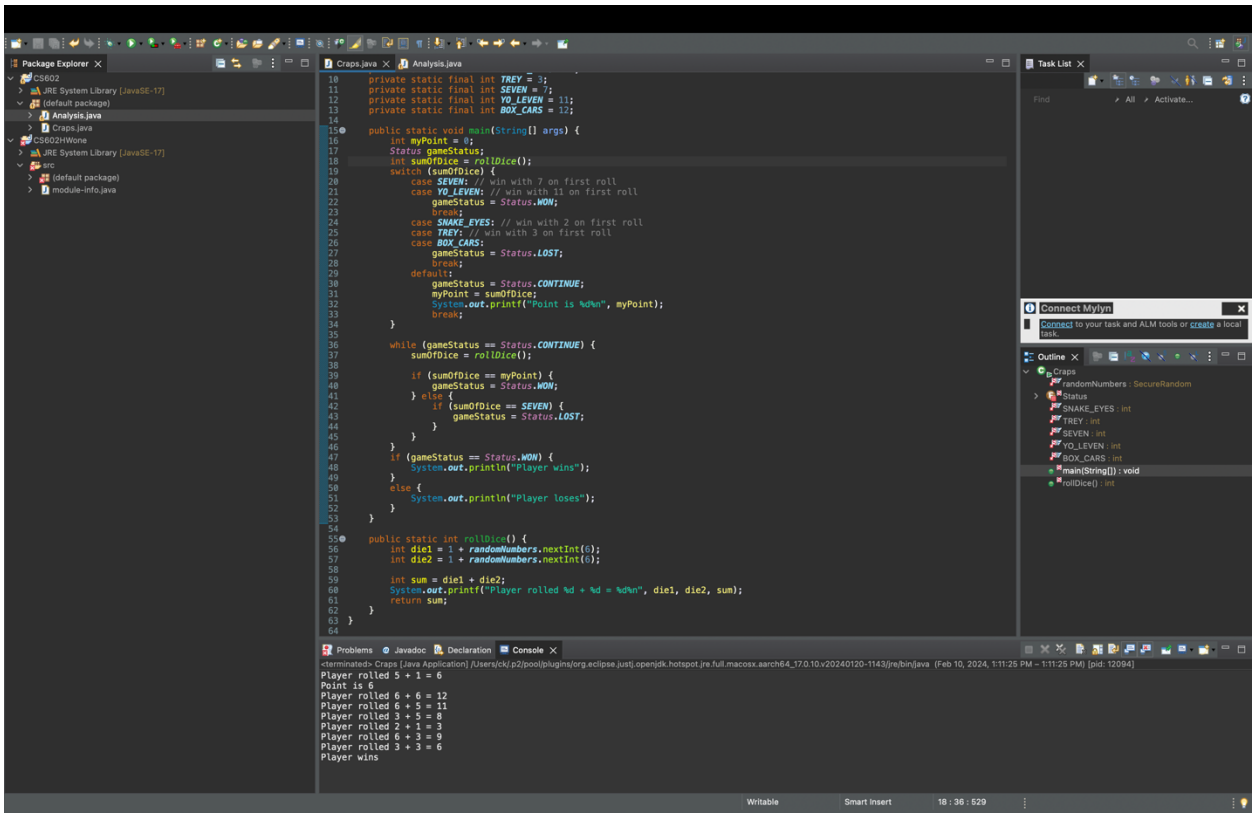
To guarantee safe random number generation, the application imports the SecureRandom class from the java.security package.

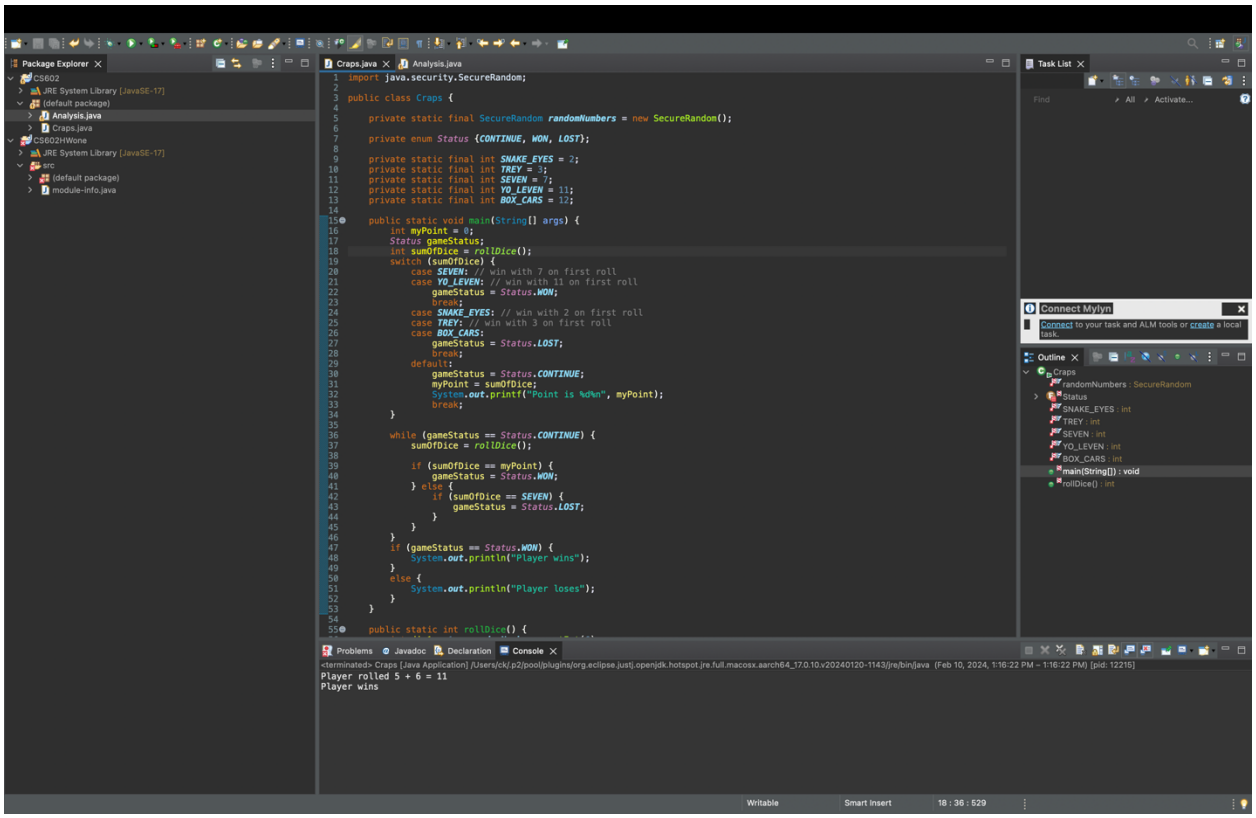
sets up variables to monitor the game status (gameStatus) and the player's point (myPoint).
computes the sum of two dice rolls using the rollDice() function.
based on the total number of dice thrown, uses a switch statement to determine the starting

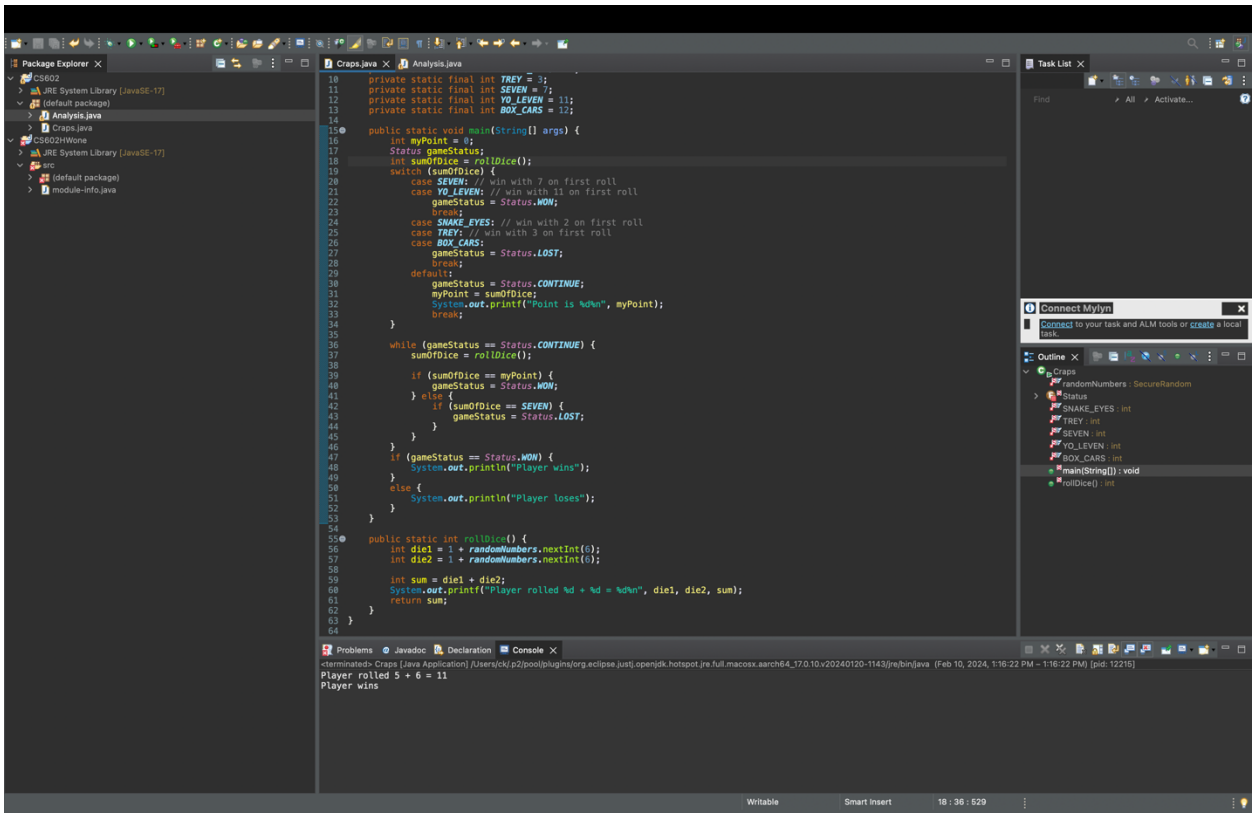
game status:
The player wins if the total is either 7 or 11.
The player is eliminated if the total is 2, 3, or 12.
If not, the player receives the sum as their point and the game goes on.
becomes stuck while the game is in the CONTINUE state:
Rolls the dice once more.
decides whether the player loses (rolling a 7) or wins (matching the point).
Prints "Player wins" or "Player loses" according to the game's ultimate status.

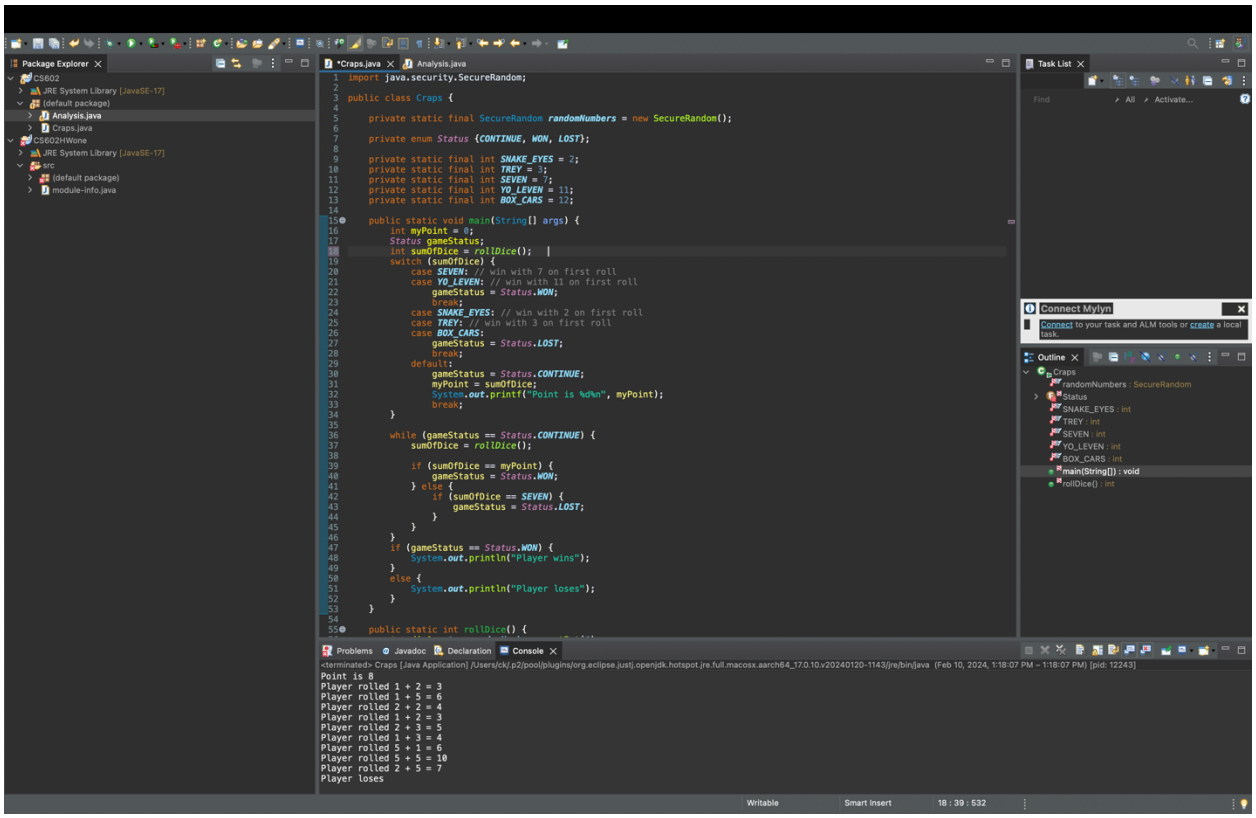
Using SecureRandom, generate random numbers for two dice rolls.
adds the two dice together.
displays the results of the dice rolls.
gives back the total.
Through the use of dice rolls simulation and the application of the game's rules to decide win or loss, this program essentially gives users the feeling of playing Craps.

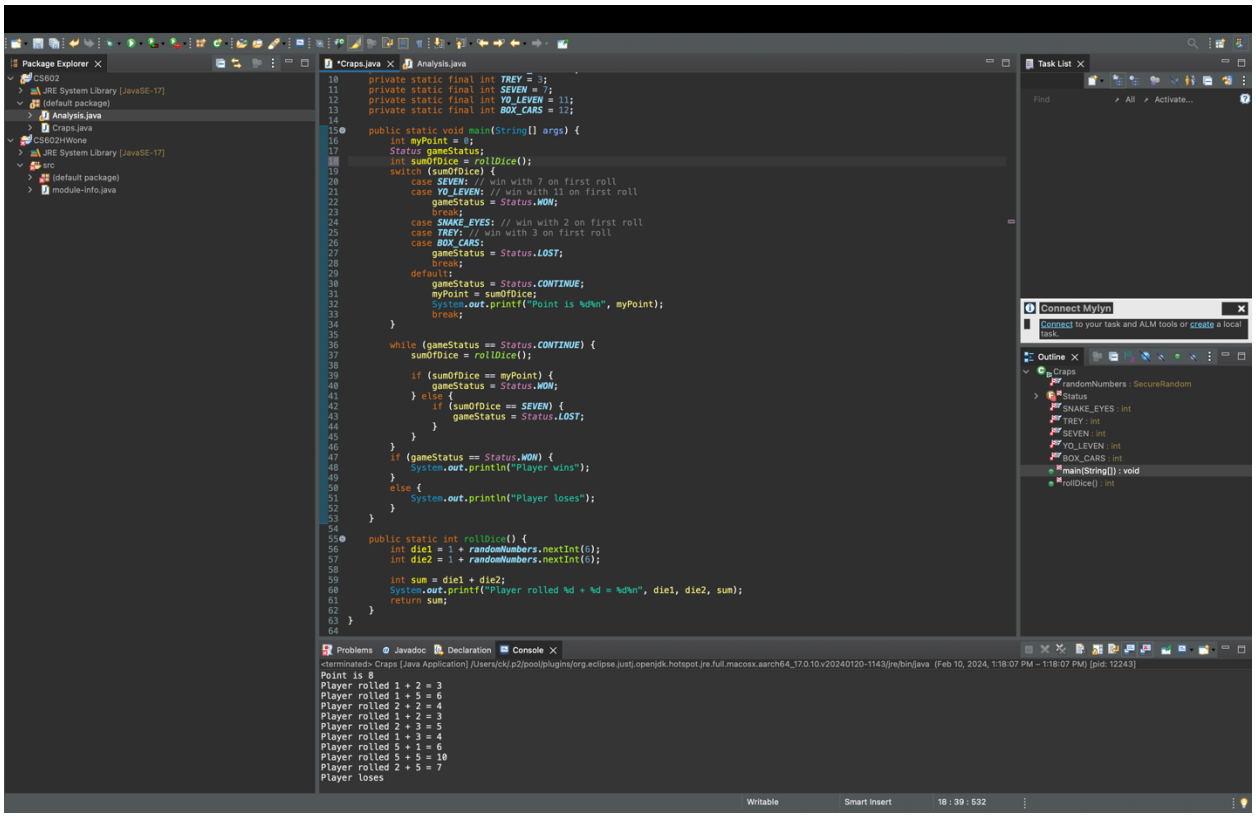


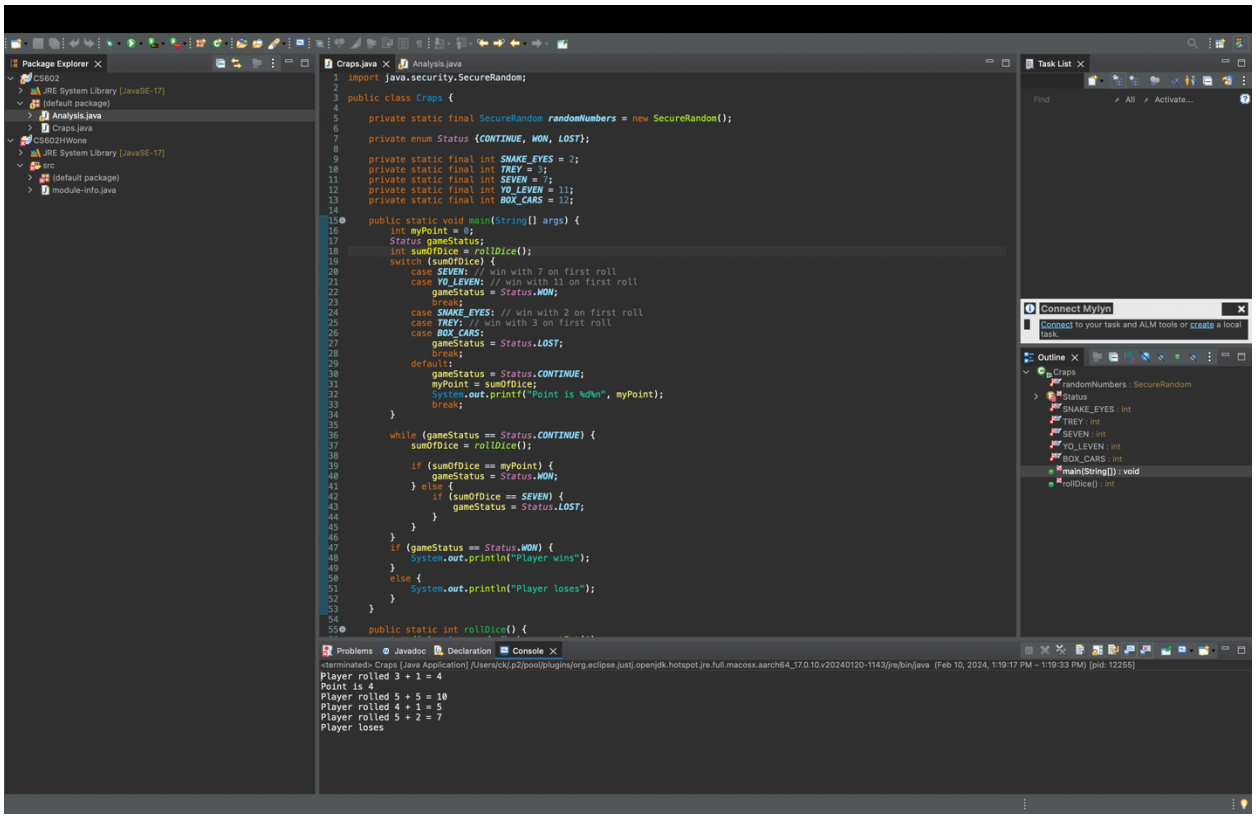

Package Explorer
CS602
JRE System Library [JavaSE-17]
(default package)
Analysis.java
Craps.java
CS602HWone
JRE System Library [JavaSE-17]
src
(default package)
module-info.java
Craps.java
Analysis.java
private static final int TREY = 3;
private static final int SEVEN = 7;
private static final int YO_LEVEN = 11;
private static final int BOX_CARS = 12;
public static void main(String[] args) {
int myPoint = 0;
Status gameStatus;
int sumOfDice = rollDice();
switch (sumOfDice) {
case SEVEN: // win with 7 on first roll
case YO_LEVEN: // win with 11 on first roll
gameStatus = Status.WON;
break;
case SNAKE_EYES: // win with 2 on first roll
case TREY: // win with 3 on first roll
case BOX_CARS:
gameStatus = Status.LOST;
break;
default:
gameStatus = Status.CONTINUE;
myPoint = sumOfDice;
System.out.printf("Point is %d%n", myPoint);
break;
}
while (gameStatus == Status.CONTINUE) {
sumOfDice = rollDice();
if (sumOfDice == myPoint) {
gameStatus = Status.WON;
} else {
if (sumOfDice == SEVEN) {
gameStatus = Status.LOST;
}
}
}
if (gameStatus == Status.WON) {
System.out.println("Player wins");
}
else {
System.out.println("Player loses");
}
}
public static int rollDice() {
int die1 = 1 + randomNumbers.nextInt(6);
int die2 = 1 + randomNumbers.nextInt(6);
int sum = die1 + die2;
System.out.printf("Player rolled %d + %d = %d%n", die1, die2, sum);
return sum;
}
}
Task List
Connect Mylyn
Connect to your task and ALM tools or create a local task.
Outline
Craps
randomNumbers : SecureRandom
Status
SNAKE_EYES : int
TREY : int
SEVEN : int
YO_LEVEN : int
BOX_CARS : int
main(String[]) : void
rollDice() : int
Problems
Javadoc
Declaration
Console
<terminated> Craps [Java Application] /Users/ck/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.aarch64_17.0.10.v20240120-1143/jre/bin/java (Feb 10, 2024, 1:11:25 PM – 1:11:25 PM) [pid: 12094]
Player rolled 5 + 1 = 6
Point is 6
Player rolled 6 + 6 = 12
Player rolled 6 + 5 = 11
Player rolled 3 + 5 = 8
Player rolled 2 + 1 = 3
Player rolled 6 + 3 = 9
Player rolled 3 + 3 = 6
Player wins
Writable
Smart Insert
18 : 36 : 529

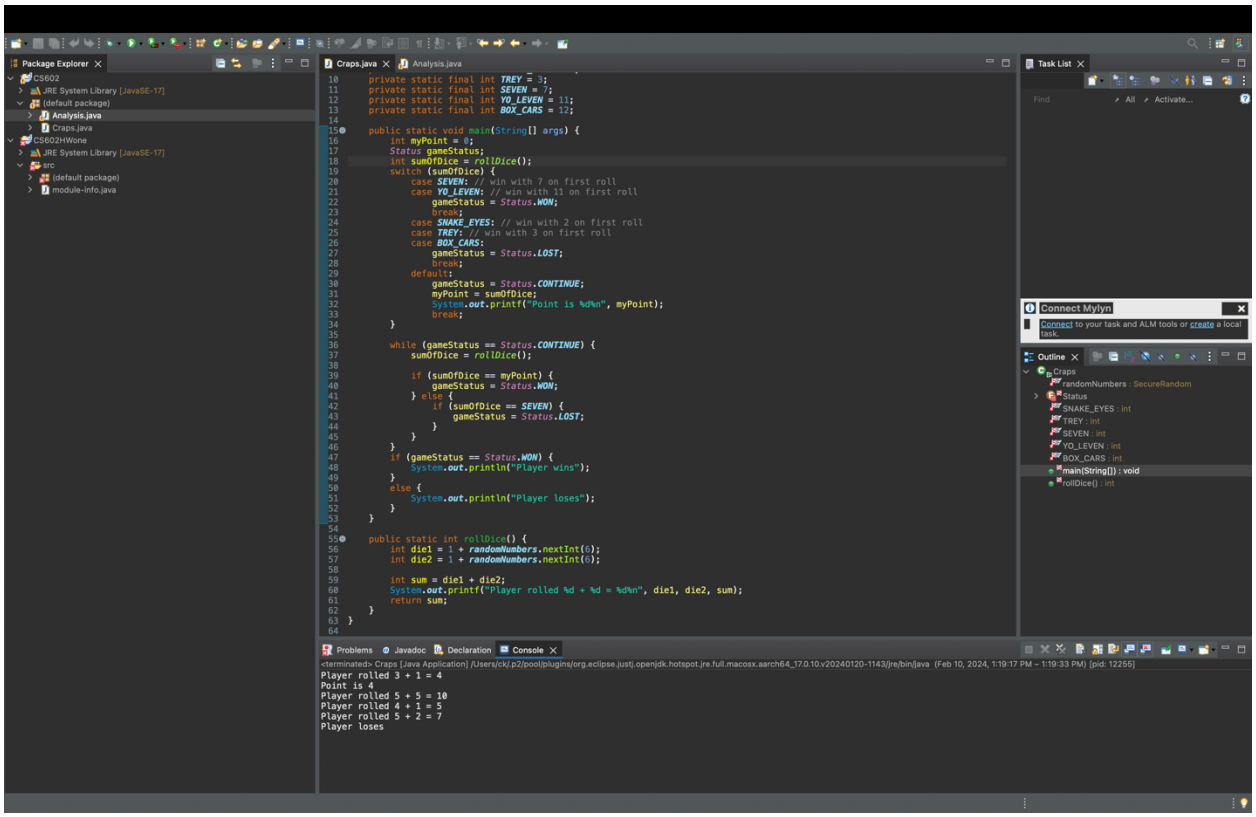

Package Explorer
CS602
JRE System Library [JavaSE-17]
(default package)
Analysis.java
Craps.java
CS602HWone
JRE System Library [JavaSE-17]
src
(default package)
module-info.java
Craps.java
Analysis.java
import java.security.SecureRandom;
public class Craps {
private static final SecureRandom randomNumbers = new SecureRandom();
private enum Status {CONTINUE, WON, LOST};
private static final int SNAKE_EYES = 2;
private static final int TREY = 3;
private static final int SEVEN = 7;
private static final int YO_LEVEN = 11;
private static final int BOX_CARS = 12;
public static void main(String[] args) {
int myPoint = 0;
Status gameStatus;
int sumOfDice = rollDice();
switch (sumOfDice) {
case SEVEN: // win with 7 on first roll
case YO_LEVEN: // win with 11 on first roll
gameStatus = Status.WON;
break;
case SNAKE_EYES: // win with 2 on first roll
case TREY: // win with 3 on first roll
case BOX_CARS:
gameStatus = Status.LOST;
break;
default:
gameStatus = Status.CONTINUE;
myPoint = sumOfDice;
System.out.printf("Point is %d%n", myPoint);
break;
}
while (gameStatus == Status.CONTINUE) {
sumOfDice = rollDice();
if (sumOfDice == myPoint) {
gameStatus = Status.WON;
} else {
if (sumOfDice == SEVEN) {
gameStatus = Status.LOST;
}
}
}
if (gameStatus == Status.WON) {
System.out.println("Player wins");
}
else {
System.out.println("Player loses");
}
}
public static int rollDice() {
Task List
Connect Mylyn
Connect to your task and ALM tools or create a local task.
Outline
Craps
randomNumbers : SecureRandom
Status
SNAKE_EYES : int
TREY : int
SEVEN : int
YO_LEVEN : int
BOX_CARS : int
main(String[]) : void
rollDice() : int
Problems
Javadoc
Declaration
Console
<terminated> Craps [Java Application] /Users/ck/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.aarch64_17.0.10.v20240120-1143/jre/bin/java (Feb 10, 2024, 1:16:22 PM – 1:16:22 PM) [pid: 12215]
Player rolled 5 + 6 = 11
Player wins
Writable
Smart Insert
18 : 36 : 529


Package Explorer
CS602
JRE System Library [JavaSE-17]
(default package)
Analysis.java
Craps.java
CS602HWone
JRE System Library [JavaSE-17]
src
(default package)
module-info.java
Craps.java
Analysis.java
private static final int TREY = 3;
private static final int SEVEN = 7;
private static final int YO_LEVEN = 11;
private static final int BOX_CARS = 12;
public static void main(String[] args) {
int myPoint = 0;
Status gameStatus;
int sumOfDice = rollDice();
switch (sumOfDice) {
case SEVEN: // win with 7 on first roll
case YO_LEVEN: // win with 11 on first roll
gameStatus = Status.WON;
break;
case SNAKE_EYES: // win with 2 on first roll
case TREY: // win with 3 on first roll
case BOX_CARS:
gameStatus = Status.LOST;
break;
default:
gameStatus = Status.CONTINUE;
myPoint = sumOfDice;
System.out.printf("Point is %d%n", myPoint);
break;
}
while (gameStatus == Status.CONTINUE) {
sumOfDice = rollDice();
if (sumOfDice == myPoint) {
gameStatus = Status.WON;
} else {
if (sumOfDice == SEVEN) {
gameStatus = Status.LOST;
}
}
}
if (gameStatus == Status.WON) {
System.out.println("Player wins");
}
else {
System.out.println("Player loses");
}
}
public static int rollDice() {
int die1 = 1 + randomNumbers.nextInt(6);
int die2 = 1 + randomNumbers.nextInt(6);
int sum = die1 + die2;
System.out.printf("Player rolled %d + %d = %d%n", die1, die2, sum);
return sum;
}
}
Task List
Connect Mylyn
Connect to your task and ALM tools or create a local task.
Outline
Craps
randomNumbers : SecureRandom
Status
SNAKE_EYES : int
TREY : int
SEVEN : int
YO_LEVEN : int
BOX_CARS : int
main(String[]) : void
rollDice() : int
Problems
Javadoc
Declaration
Console
<terminated> Craps [Java Application] /Users/ck/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.aarch64_17.0.10.v20240120-1143/jre/bin/java (Feb 10, 2024, 1:16:22 PM – 1:16:22 PM) [pid: 12215]
Player rolled 5 + 6 = 11
Player wins
Writable
Smart Insert
18 : 36 : 529


Package Explorer
CS602
JRE System Library [JavaSE-17]
(default package)
Analysis.java
Craps.java
CS602HWone
JRE System Library [JavaSE-17]
src
(default package)
module-info.java
*Craps.java
Analysis.java
import java.security.SecureRandom;
public class Craps {
private static final SecureRandom randomNumbers = new SecureRandom();
private enum Status {CONTINUE, WON, LOST};
private static final int SNAKE_EYES = 2;
private static final int TREY = 3;
private static final int SEVEN = 7;
private static final int YO_LEVEN = 11;
private static final int BOX_CARS = 12;
public static void main(String[] args) {
int myPoint = 0;
Status gameStatus;
int sumOfDice = rollDice();
switch (sumOfDice) {
case SEVEN: // win with 7 on first roll
case YO_LEVEN: // win with 11 on first roll
gameStatus = Status.WON;
break;
case SNAKE_EYES: // win with 2 on first roll
case TREY: // win with 3 on first roll
case BOX_CARS:
gameStatus = Status.LOST;
break;
default:
gameStatus = Status.CONTINUE;
myPoint = sumOfDice;
System.out.printf("Point is %d%n", myPoint);
break;
}
while (gameStatus == Status.CONTINUE) {
sumOfDice = rollDice();
if (sumOfDice == myPoint) {
gameStatus = Status.WON;
} else {
if (sumOfDice == SEVEN) {
gameStatus = Status.LOST;
}
}
}
if (gameStatus == Status.WON) {
System.out.println("Player wins");
}
else {
System.out.println("Player loses");
}
}
public static int rollDice() {
Task List
Connect Mylyn
Connect to your task and ALM tools or create a local task.
Outline
Craps
randomNumbers : SecureRandom
Status
SNAKE_EYES : int
TREY : int
SEVEN : int
YO_LEVEN : int
BOX_CARS : int
main(String[]) : void
rollDice() : int
Problems
Javadoc
Declaration
Console
<terminated> Craps [Java Application] /Users/ck/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.aarch64_17.0.10.v20240120-1143/jre/bin/java (Feb 10, 2024, 1:18:07 PM – 1:18:07 PM) [pid: 12243]
Point is 8
Player rolled 1 + 2 = 3
Player rolled 1 + 5 = 6
Player rolled 2 + 2 = 4
Player rolled 1 + 2 = 3
Player rolled 2 + 3 = 5
Player rolled 1 + 3 = 4
Player rolled 5 + 1 = 6
Player rolled 5 + 5 = 10
Player rolled 2 + 5 = 7
Player loses
Writable
Smart Insert
18 : 39 : 532


Package Explorer
CS602
JRE System Library [JavaSE-17]
(default package)
Analysis.java
Craps.java
CS602HWone
JRE System Library [JavaSE-17]
src
(default package)
module-info.java
*Craps.java
Analysis.java
Task List
Connect Mylyn
Connect to your task and ALM tools or create a local task.
Outline
Craps
randomNumbers : SecureRandom
Status
SNAKE_EYES : int
TREY : int
SEVEN : int
YO_LEVEN : int
BOX_CARS : int
main(String[]) : void
rollDice() : int
Problems
Javadoc
Declaration
Console
<terminated> Craps [Java Application] /Users/ck/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.aarch64_17.0.10.v20240120-1143/jre/bin/java (Feb 10, 2024, 1:18:07 PM – 1:18:07 PM) [pid: 12243]
Point is 8
Player rolled 1 + 2 = 3
Player rolled 1 + 5 = 6
Player rolled 2 + 2 = 4
Player rolled 1 + 2 = 3
Player rolled 2 + 3 = 5
Player rolled 1 + 3 = 4
Player rolled 5 + 1 = 6
Player rolled 5 + 5 = 10
Player rolled 2 + 5 = 7
Player loses
Writable
Smart Insert
18 : 39 : 532


Package Explorer
CS602
JRE System Library [JavaSE-17]
(default package)
Analysis.java
Craps.java
CS602HWone
JRE System Library [JavaSE-17]
src
(default package)
module-info.java
Craps.java
Analysis.java
Task List
Connect Mylyn
Connect to your task and ALM tools or create a local task.
Outline
Craps
randomNumbers : SecureRandom
Status
SNAKE_EYES : int
TREY : int
SEVEN : int
YO_LEVEN : int
BOX_CARS : int
main(String[]) : void
rollDice() : int
Problems
Javadoc
Declaration
Console
<terminated> Craps [Java Application] /Users/ck/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.aarch64_17.0.10.v20240120-1143/jre/bin/java (Feb 10, 2024, 1:19:17 PM – 1:19:33 PM) [pid: 12255]
Player rolled 3 + 1 = 4
Point is 4
Player rolled 5 + 5 = 10
Player rolled 4 + 1 = 5
Player rolled 5 + 2 = 7
Player loses



f. How GradeBookTest.java works, pages 274-279 (1 point)

Answer:

Description:

GradeBookTest, a Java application, is a tool for processing and evaluating grades. Here is a rundown of its main elements and features:

The course name and the students' grades are represented by the instance variables courseName and grades, respectively, in the class.
The supplied course name and grades array are used by the constructor to initialize these variables.
There are established public procedures for modifying and examining the grade data.

Method: getCourseName()
gives back the course name.

Procedure: processGrades()
produces each student's average and grades.
determines and shows the grades in the gradebook that are lowest and highest.
creates a bar chart that shows the grade distribution.

getMaximum() and getMinimum() Methods:

These methods calculate the gradebook's lowest and highest grades, respectively.
The getAverage() Method

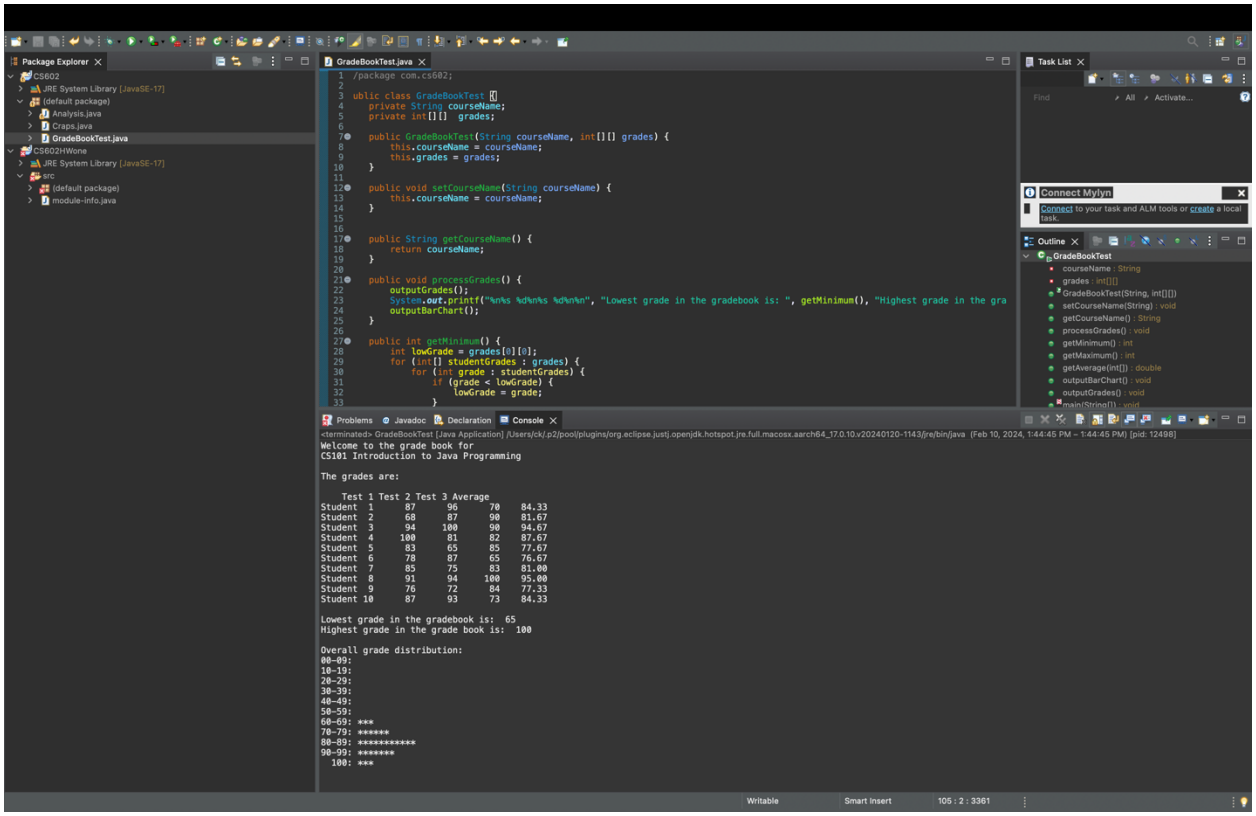
determines the average grade for a specific group of grades.
resultsUsing BarChart():

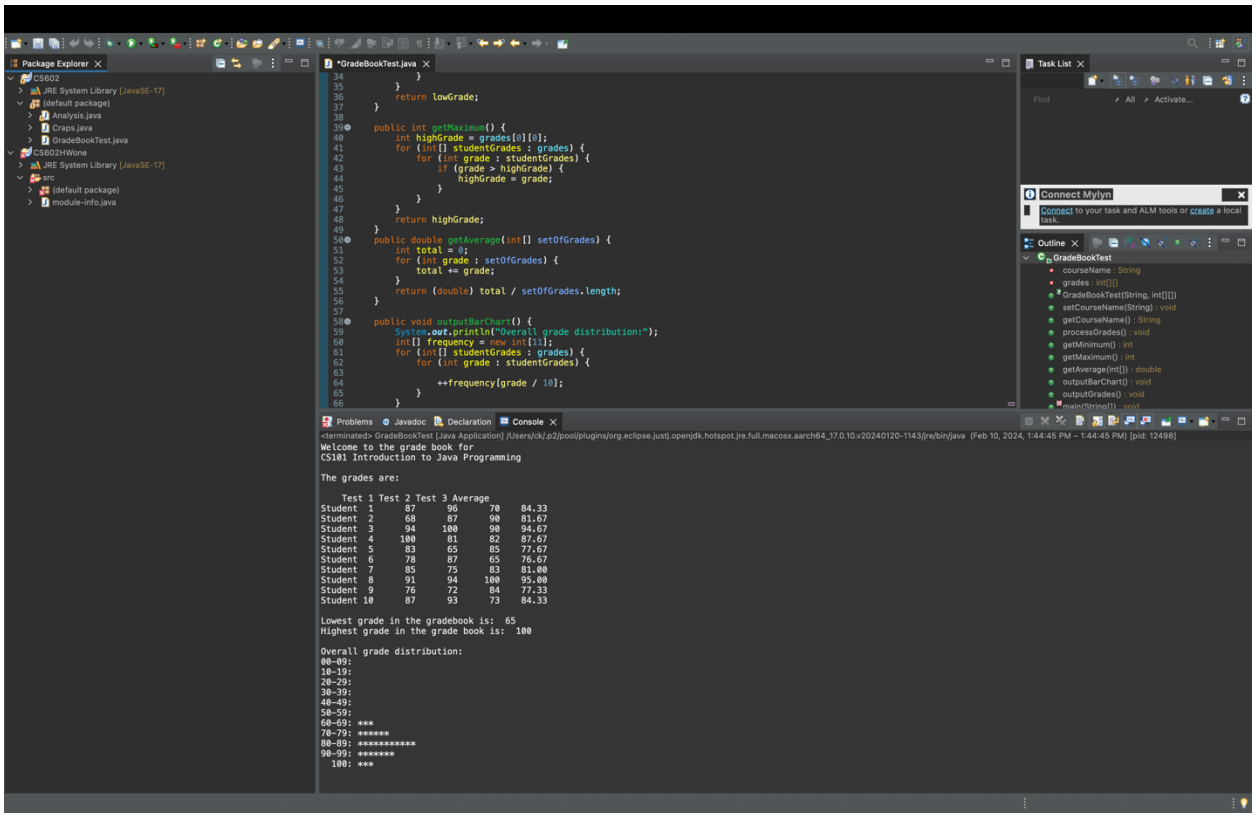
Using asterisks (*), creates a graphic depiction of the grade distribution.

method for outputGrades():

prints all of the student grades, including the average for every student.

assigns a specified course name and grades array to a newly created GradeBookTest instance.
presents a greeting along with the course name.
uses the processGrades() function to examine and show the grades.
With regard to a particular course, this program offers features for processing grades, calculating statistics, and visualizing grade distributions. It helps teachers efficiently manage assignment grading and provides insights into student performance.





```
        }
        return lowGrade;
    }

    public int getMaximum() {
        int highGrade = grades[0][0];
        for (int[] studentGrades : grades) {
            for (int grade : studentGrades) {
                if (grade > highGrade) {
                    highGrade = grade;
                }
            }
        }
        return highGrade;
    }
    public double getAverage(int[] setOfGrades) {
        int total = 0;
        for (int grade : setOfGrades) {
            total += grade;
        }
        return (double) total / setOfGrades.length;
    }

    public void outputBarChart() {
        System.out.println("Overall grade distribution:");
        int[] frequency = new int[11];
        for (int[] studentGrades : grades) {
            for (int grade : studentGrades) {

                ++frequency[grade / 10];
            }
        }
```

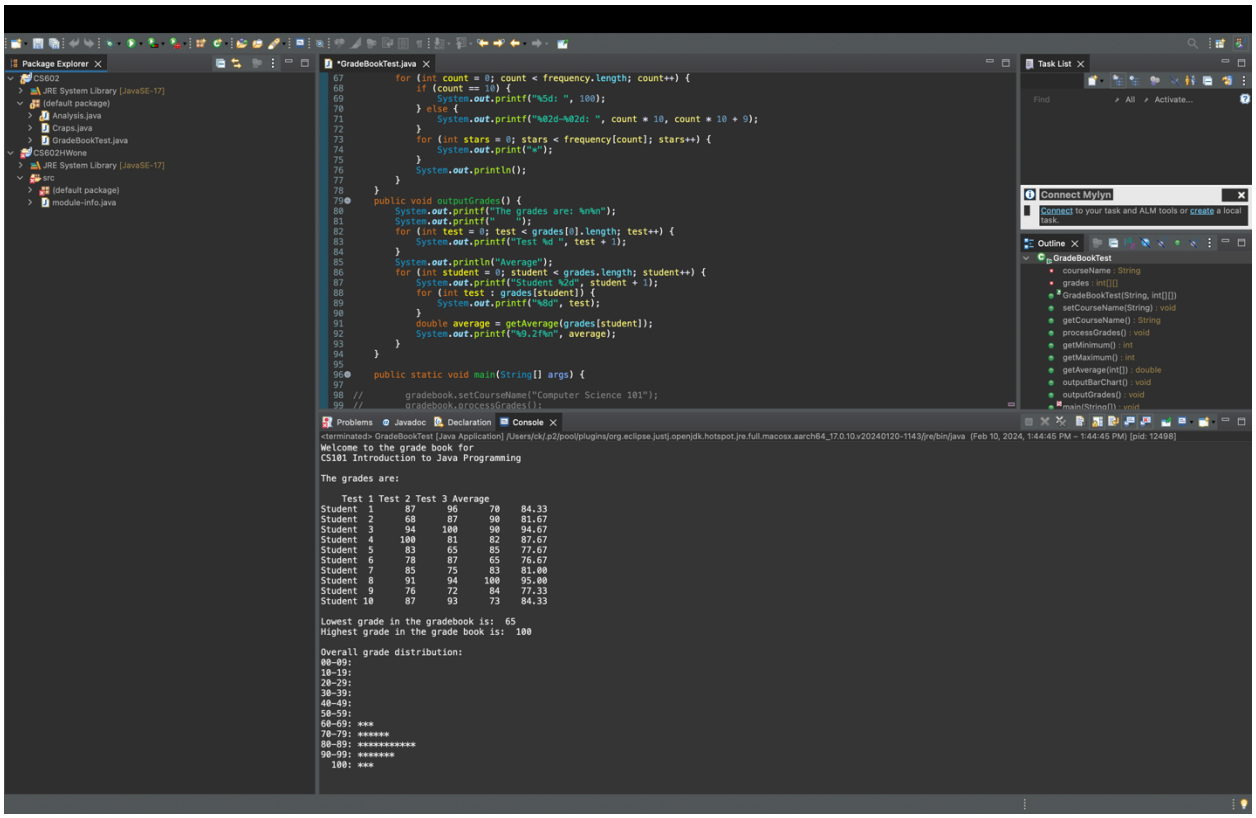
```
Welcome to the grade book for
CS101 Introduction to Java Programming

The grades are:

    Test 1 Test 2 Test 3 Average
Student  1      87      96      70    84.33
Student  2      68      87      90    81.67
Student  3      94     100      90    94.67
Student  4     100      81      82    87.67
Student  5      83      65      85    77.67
Student  6      78      87      65    76.67
Student  7      85      75      83    81.00
Student  8      91      94     100    95.00
Student  9      76      72      84    77.33
Student 10      87      93      73    84.33

Lowest grade in the gradebook is:  65
Highest grade in the grade book is:  100

Overall grade distribution:
00-09:
10-19:
20-29:
30-39:
40-49:
50-59:
60-69: ***
70-79: ******
80-89: **********
90-99: *******
  100: ***
```



```
        for (int count = 0; count < frequency.length; count++) {
            if (count == 10) {
                System.out.printf("%5d: ", 100);
            } else {
                System.out.printf("%02d-%02d: ", count * 10, count * 10 + 9);
            }
            for (int stars = 0; stars < frequency[count]; stars++) {
                System.out.print("*");
            }
            System.out.println();
        }
    }
    public void outputGrades() {
        System.out.printf("The grades are: %n%n");
        System.out.printf("            ");
        for (int test = 0; test < grades[0].length; test++) {
            System.out.printf("Test %d ", test + 1);
        }
        System.out.println("Average");
        for (int student = 0; student < grades.length; student++) {
            System.out.printf("Student %2d", student + 1);
            for (int test : grades[student]) {
                System.out.printf("%8d", test);
            }
            double average = getAverage(grades[student]);
            System.out.printf("%9.2f%n", average);
        }
    }

    public static void main(String[] args) {

//        gradebook.setCourseName("Computer Science 101");
//        gradebook.processGrades();
```

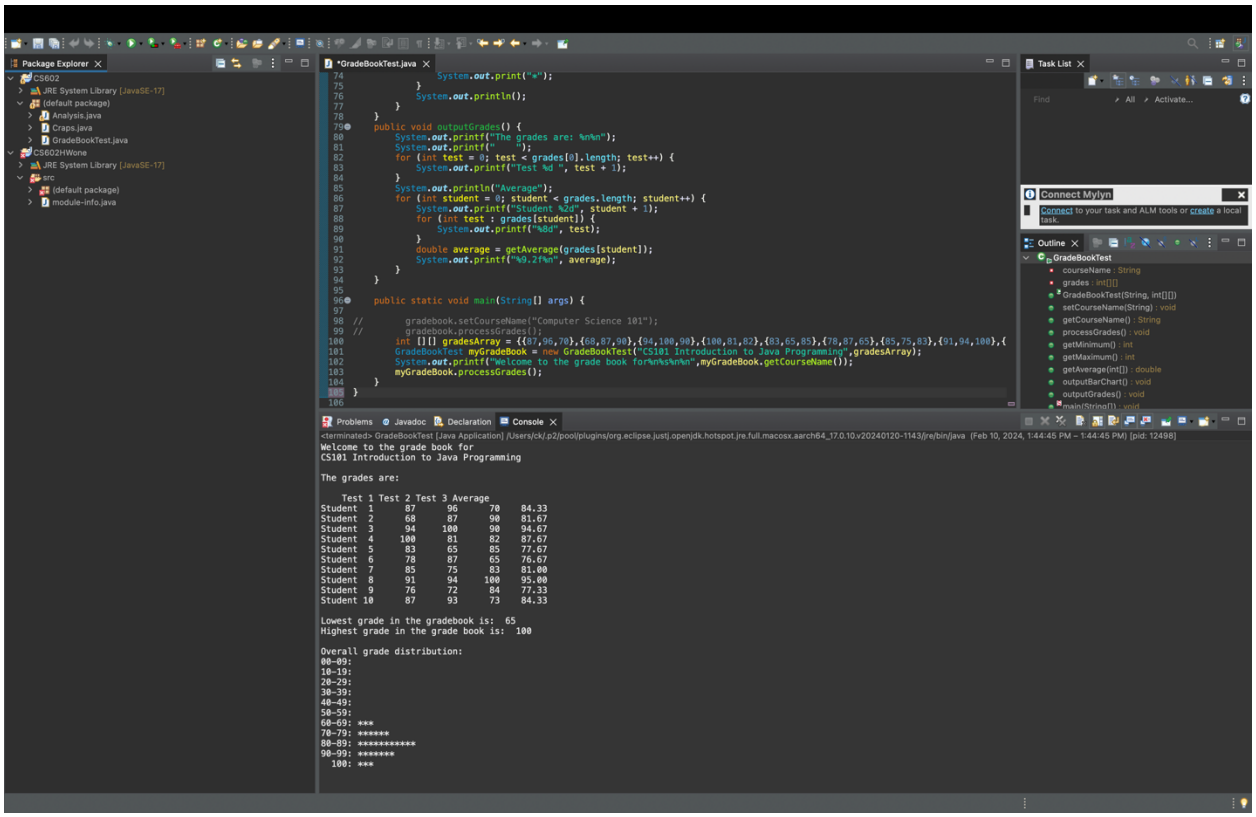
```
Welcome to the grade book for
CS101 Introduction to Java Programming

The grades are:

    Test 1 Test 2 Test 3 Average
Student  1      87      96      70    84.33
Student  2      68      87      90    81.67
Student  3      94     100      90    94.67
Student  4     100      81      82    87.67
Student  5      83      65      85    77.67
Student  6      78      87      65    76.67
Student  7      85      75      83    81.00
Student  8      91      94     100    95.00
Student  9      76      72      84    77.33
Student 10      87      93      73    84.33

Lowest grade in the gradebook is:  65
Highest grade in the grade book is:  100

Overall grade distribution:
00-09:
10-19:
20-29:
30-39:
40-49:
50-59:
60-69: ***
70-79: ******
80-89: **********
90-99: *******
  100: ***
```


Package Explorer
CS602
JRE System Library [JavaSE-17]
(default package)
Analysis.java
Craps.java
GradeBookTest.java
CS602HWone
JRE System Library [JavaSE-17]
src
(default package)
module-info.java
*GradeBookTest.java
System.out.print("*");
}
System.out.println();
}
}
public void outputGrades() {
System.out.printf("The grades are: %n%n");
System.out.printf("   ");
for (int test = 0; test < grades[0].length; test++) {
System.out.printf("Test %d ", test + 1);
}
System.out.println("Average");
for (int student = 0; student < grades.length; student++) {
System.out.printf("Student %2d", student + 1);
for (int test : grades[student]) {
System.out.printf("%8d", test);
}
double average = getAverage(grades[student]);
System.out.printf("%9.2f%n", average);
}
}
public static void main(String[] args) {
// gradebook.setCourseName("Computer Science 101");
// gradebook.processGrades();
int [][] gradesArray = {{87,96,70},{68,87,90},{94,100,90},{100,81,82},{83,65,85},{78,87,65},{85,75,83},{91,94,100},{
GradeBookTest myGradeBook = new GradeBookTest("CS101 Introduction to Java Programming",gradesArray);
System.out.printf("Welcome to the grade book for%n%s%n%n",myGradeBook.getCourseName());
myGradeBook.processGrades();
}
}
Task List
Find
All
Activate...
Connect Mylyn
Connect to your task and ALM tools or create a local task.
Outline
GradeBookTest
courseName : String
grades : int[][]
GradeBookTest(String, int[][])
setCourseName(String) : void
getCourseName() : String
processGrades() : void
getMinimum() : int
getMaximum() : int
getAverage(int[]) : double
outputBarChart() : void
outputGrades() : void
main(String[]) : void
Problems
Javadoc
Declaration
Console
<terminated> GradeBookTest [Java Application] /Users/ck/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.aarch64_17.0.10.v20240120-1143/jre/bin/java (Feb 10, 2024, 1:44:45 PM – 1:44:45 PM) [pid: 12498]
Welcome to the grade book for
CS101 Introduction to Java Programming
The grades are:
Test 1 Test 2 Test 3 Average
Student 1 87 96 70 84.33
Student 2 68 87 90 81.67
Student 3 94 100 90 94.67
Student 4 100 81 82 87.67
Student 5 83 65 85 77.67
Student 6 78 87 65 76.67
Student 7 85 75 83 81.00
Student 8 91 94 100 95.00
Student 9 76 72 84 77.33
Student 10 87 93 73 84.33
Lowest grade in the gradebook is: 65
Highest grade in the grade book is: 100
Overall grade distribution:
00-09:
10-19:
20-29:
30-39:
40-49:
50-59:
60-69: ***
70-79: ******
80-89: **********
90-99: *******
100: ***