STRAYER UNIVERSITY

CIS 406 – Java Programming I

(Prerequisite: CIS 110)

COURSE DESCRIPTION

This course introduces students to the fundamental constructs of the Java object-oriented programming language. Students will test, document, and design business-oriented programs. Topics include objects, classes, iteration, encapsulation, polymorphism, and inheritance.

INSTRUCTIONAL MATERIALS

Required Resources

Farrell, J., (2014). *Java Programming I* (7th ed.). Independence, KY: Cengage.

> **Note:** This Strayer custom text is excerpted from: Farrell, J. (2014). *Java Programming* (7th ed.).

Eclipse Integrated Development Environment (IDE) is recommended by Strayer University to its students and faculty as the preferred IDE for this course. Further information on the free download of the Eclipse IDE, alternative IDEs, restrictions, and tutorials can be found under *Software* in the Notes section of this Course Guide.

Supplemental Resources

Eckel, B. (2002). *Thinking in Java, 3rd edition*. Upper Saddle River, NJ: Prentice Hall.

Gau, T. (2011). *Java video tutorials*. Retrieved from http://www.javavideotutes.com/

Java Beginner. (2008). General format. Retrieved from http://www.javabeginner.com/

The Java Tutorials. (2012). General format. Retrieved from http://docs.oracle.com/javase/tutorial/

COURSE LEARNING OUTCOMES

1. Demonstrate the proper use and application of syntax in the Java programming language.
2. Demonstrate the ability to design, compile, implement, test, and debug simple programs in Java.
3. Demonstrate the ability to manipulate numbers and character strings in Java.
4. Compare and contrast classes and objects in Java.
5. Construct classes through systematic procedures.
6. Discuss object-oriented design principles.
7. Compare and contrast abstract and concrete data types.
8. Demonstrate the ability to implement generic classes and methods.
9. Differentiate between static and non-static methods and variables.
10. Declare and use interface types.
11. Demonstrate the ability to program simple and complex decisions in Java.
12. Describe and implement the bubble sort algorithm.
13. Implement loops for repetitive tasks.
14. Compare and contrast definite loops and indefinite loops.
15. Compare and contrast arrays and array lists in Java.
16. Describe and implement iterators in Java.
17. Discuss the concepts of inheritance and polymorphism.

18. Use technology and information resources to research issues in Java programming.
19. Write clearly and concisely about Java programming using proper writing mechanics and technical style conventions.

WEEKLY COURSE SCHEDULE

The standard requirement for a 4.5 credit hour course is for students to spend 13.5 hours in weekly work. This includes preparation, activities, and evaluation regardless of delivery mode.

Week	Preparation, Activities, and Evaluation	Points
1	Preparation • Reading(s) ○ Chapter 1: Creating Java Programs • Tutorial ○ The students should access and use the following tutorials prior to completing the lab assignments for this lesson. A set of tutorials and information on the use of the Eclipse Integrated Development Environment (IDE) is located at http://eclipsetutorial.sourceforge.net/totalbeginner.html. ○ Further information on the free download of the Eclipse IDE, alternative IDEs, restrictions, and tutorials can be found under *Software* in the Notes section of this Course Guide. • Video(s), accessible in the online course shell ○ “Object-Oriented Programming” (4 min 13 s) ○ “A Java Program” (3 min 22 s) ○ “Compiling and Executing a Program” (3 min 41 s) Activities • Discussion Evaluation • None	 20
2	Preparation • Reading(s) ○ Chapter 2: Using Data • Video(s), accessible in the online course shell ○ “Declaring Variables and Constants” (4 min 13 s) ○ “Getting Input” (5 min 22 s) ○ “Arithmetic” (4 min 02 s) Activities • Discussion Evaluation • Lab 1: Exercise 4 on page 113	 20 20
3	Preparation • Reading(s) ○ Chapter 3: Using Methods, Classes, and Objects	

	• Video(s), accessible in the online course shell ○ "Methods" (1 min 44 s) ○ "Methods and Parameters" (4 min 09 s) ○ "Classes and Objects" (2 min 12 s) ○ "Constructors" (3 min 11 s) Activities • Discussion Evaluation • Lab 2: Exercise 3 on page 172	20 20
4	Preparation • Reading(s) ○ Chapter 4: More Object Concepts • Video(s), accessible in the online course shell ○ "Overloading Constructors" (4 min 28 s) ○ "The 'this' Reference" (4 min 34 s) Activities • Discussion Evaluation • Lab 3: Exercise 2 on page 235	20 20
5	Preparation • Reading(s) ○ Chapter 5: Making Decisions • Video(s), accessible in the online course shell ○ "Making Decisions" (3 min 53 s) ○ "Using && and II" (2 min 42 s) ○ "Using the 'switch' Statement" (3 min 53 s) Activities • Discussion Evaluation • Midterm Exam: Chapters 1 through 4	20 100
6	Preparation • Reading(s) ○ Chapter 6: Looping • Video(s), accessible in the online course shell ○ "Looping" (3 min 34 s) ○ "Using Shortcut Arithmetic Operators" (4 min 30 s) ○ "Using the for Loop" (2 min 00 s) ○ "Nested Loops" (5 min 35 s) Activities • Discussion Evaluation • Assignment 1: Financial Portfolio	20 200

7	Preparation • Reading(s) ○ Chapter 7: Characters, Strings, and the StringBuilder • Video(s), accessible in the online course shell ○ "Comparing Strings" (4 min 43 s) ○ "String Methods" (5 min 03 s) ○ "StringBuilder" (3 min 05 s) Activities • Discussion Evaluation • Lab 4: Exercise 1 on page 387	 20 20
8	Preparation • Reading(s) ○ Chapter 8: Arrays • Video(s), accessible in the online course shell ○ "Arrays" (2 min 59 s) ○ "Searching an Array" (3 min 24 s) ○ "Arrays and Methods" (2 min 33 s) Activities • Discussion Evaluation • Case Study: Business Applications of Java	 20 100
9	Preparation • Reading(s) ○ Chapter 9: Advanced Array Concepts • Video(s), accessible in the online course shell ○ "Sorting" (6 min 10 s) ○ "The Insertion Sort" (2 min 34 s) ○ "Two-Dimensional Arrays" (2 min 15 s) ○ "Enumerations" (4 min 04 s) Activities • Discussion Evaluation • Lab 5: Exercise 1 on page 492	 20 20
10	Preparation • Reading(s) ○ Read the article titled, "What Is Inheritance?" located at http://docs.oracle.com/javase/tutorial/java/concepts/inheritance.html. ○ Read the article titled, "Inheritance", located at http://docs.oracle.com/javase/tutorial/java/IandI/subclasses.html ○ Read the article titled, "Polymorphism", located at http://docs.oracle.com/javase/tutorial/java/IandI/polymorphism.h	

	tml. • Videos ○ Watch the video titled "Java Programming Tutorial - 49 - Inheritance" (9 min 24 s). Video Source: thenewboston. (2009, Aug 9). Java Programming Tutorial - 49 - Inheritance [Video file]. Retrieved from http://www.youtube.com/watch?v=9JpNY-XAseg. This video can be viewed from within your online course shell. ○ Watch the video titled "Java Programming Tutorial - 55 - Intoduction to Polymorphism" (8 min 20 s). Video Source: thenewboston. (2009, Sep 23). Java Programming Tutorial - 55 - Intoduction to Polymorphism [Video file]. Retrieved http://www.youtube.com/watch?v=0xw06loTm1k. This video can be viewed from within your online course shell. Activities • Discussion Evaluation • Assignment 2: uGrade	20 200
11	Preparation • Reading(s): None Activities • Discussion Evaluation • Final Exam: Chapters 5 through 9	100

GRADING SCALE – UNDERGRADUATE

Assignment	Total Points	% of Grade
Lab Assignments (5 problems worth 20 points apiece)	100	10%
Assignment 1: Financial Portfolio	200	20%
Case Study: Business Applications of Java	100	10%
Assignment 2: uGrade	200	20%
Midterm Exam (Chapters 1-4) (open book, 2-hour time limit, with 25 multiple choice questions worth 4 points apiece)	100	10%
Final Exam (Chapters 5-9) (open book, 2-hour time limit, with 25 multiple choice questions worth 4 points apiece)	100	10%
Participation (10 discussions worth 20 points apiece) **Note:** Week 11 discussion is not graded.	200	20%
Totals	1,000	100%

Points	Percentage	Grade
900 – 1,000	90% – 100%	A
800 – 899	80% – 89%	B
700 – 799	70% – 79%	C
600 – 699	60% – 69%	D
Below 600	Below 60%	F

Writing Assignments

The objective of the School of Information Systems' writing assignments is to promote attitudes and skills that will improve a student's ability to communicate in writing, develop research skills and documentation techniques, and encourage critical analysis of data and conclusions specific to the course learning outcomes in the information systems and technology domain.

Lab Assignments

Worth 20 points apiece

Complete the weekly labs based on the following:

- Write the code for each lab assignment.
- The lab is to be submitted in a single zip file in the online course shell, which must contain all .java files, along with any additional files that may be necessary for your project to run (ex: text files).
- Any and all written answers must be entered into the online course shell with the submission of the attached lab assignment.

Weekly Lab Breakdown	
Week Due	**Graded Lab Exercises**
1	None
2	Lab 1: Exercise 4 on page 113
3	Lab 2: Exercise 3 on page 172
4	Lab 3: Exercise 2 on page 235
5	None
6	None
7	Lab 4: Exercise 1 on page 387
8	None
9	Lab 5: Exercise 1 on page 492
10	None

Each lab assignment will be graded based on the following:

1. The program must compile, execute, produce correct results, and meet all of the specifications in the weekly lab.

Additionally you must:

2. Organize the code for user readability.
3. Organize the code for reusability.
4. Provide documentation with embedded comments for reader understanding.
5. Organize the code for efficiency.

Grading for each lab assignment will be based on the following rubric.

Points: 20	**Java Programming Lab Assignment Rubric**				
Criteria	**Unacceptable Below 60% F**	**Meets Minimum Expectations 60-69% D**	**Fair 70-79% C**	**Proficient 80-89% B**	**Exemplary 90-100% A**
1. Specifications Weight: 60%	The program does not compile.	The program compiles but does not execute.	The program compiles, executes but produces incorrect results.	The program compiles, executes, produces correct results, but does not meet all of the specifications.	The program compiles, executes, produces correct results, and meets all of the specifications.
2. Readability Weight: 10%	The code is not organized and very difficult to read.	The code is poorly organized and difficult to read.	The code is partially organized but readable only by someone who knows the expected end result.	The code is organized and easy to read.	The code is exceptionally organized and very easy to read.
3. Reusability and object-oriented programming constructs Weight: 10%	The code is not organized for reusability.	The code is poorly organized for reusability.	The code is partially organized and some parts of the code could be reused in other programs.	The code is organized and most of the code could be reused in other programs.	The code is exceptionally organized and could be reused as a whole or each routine could be reused.
4. Documentation Weight: 10%	No documentation is provided.	The documentation consists of embedded comments but does not help the reader understand the code.	The documentation consists of embedded comments and some header comments separating routines.	The documentation consists of embedded comments and header documentation that is useful in understanding the code.	The documentation consists of embedded comments and clearly explains what the code is accomplishing and how.
5. Efficiency Weight: 10%	The code is unnecessarily long and appears to be patched together.	The code is unnecessarily long.	The code is fairly efficient but sacrificing readability and understanding.	The code is efficient without sacrificing readability and understanding.	The code is extremely efficient without sacrificing readability and understanding.

Assignment 1: Financial Portfolio
Due Week 6 and worth 200 points

This assignment consists of two (2) sections:

- A java program file
- A screen shot of the output and a description of your Java program

Label each file name according to the section of the assignment for which it is written. Put both sections together in a single zip file and submit the zip file.

Suppose you are a Java programmer for an investment company. Your Chief Technology Officer (CTO) has asked you to development an interactive Java application that will be used by investment advisors to analyze clients' bank portfolios. The application must show the investment advisor the total value of the assets, and the value of the individual assets (savings account, stocks investments, and bonds investments).

Section 1: Java Program File

1. Create a Java program according to the specifications stated below:
    - Include a composition class called "FinancialPortofolio"
    - Public attributes for the composition class must include the client's first name (string data type), last name (string data type), portfolio number (integer data type), and total value of the portfolio (double data type)
    - The composition class must include a savings account class called "SavingsAccount" with the following public attributes: an account number (string), and an account balance (double)
    - The composition class must include a bonds class called "Bonds" with the following public attributes: bond name (string), face value (double), and number of bonds (integer)
    - The composition class must include a stocks class called "Stocks" with the following public attributes: stock name (string), stock value (float), and number of shares (integer)
    - Create setters and getters methods for all the public attributes in each of the classes
    - Create objects that prompt the user (investment advisor) to enter all of the values for each of the classes
    - The savings account object must add the balance to the portfolio total value
    - The bonds object must add the total bonds value (bond value multiplied by the number of bonds) to the portfolio total value
    - The stocks object must add the total stocks value (stock value multiplied by the number of shares) to the total portfolio value
    - When a user (investment advisor) runs the Java program, it must prompt the advisor to enter the financial portfolio data, savings account data, stocks data, and bonds data. The program must compute the total value of the portfolio for each asset (savings account, stocks, and bonds).
    - When all of the data has been entered and the total value of the portfolio has been calculated, the program must display the results using the following format as an example:

Portfolio Name:	Jane's Portfolio
Savings account:	Blue Bank ($2000.00)
Bonds:	Derby ($3000.00)
Stocks:	IBM ($10000.00)
Portfolio value:	$15000.00

Section 2: Screen Shot of the Output and Description of Your Java Program

2. Create a screen shot of the interactive session output, and include a description of your Java program.
    - Submit a screen shot which shows the output of your Java Program. **Note:** Go to http://www.take-a-screenshot.org/ if you need a tutorial on taking a screen shot.
    - Include a one (1) page description about your program. **Note:** Use MS Word for your program description, and place the screen shot of the output from your Java program into the Word file as an attached image.

Section 1 and Section 2 will be graded based on the following:

1. The program must compile, execute, produce correct results, and meet all of the specifications stated in Section 1.

Additionally you must:

2. Organize the code for user readability.
3. Organize the code for reusability.
4. Organize the code for efficiency.
5. Provide documentation with embedded comments for reader understanding.
6. Include a one (1) page description about your program.

The specific course learning outcomes associated with this assignment are:

- Demonstrate the proper use and application of syntax in the Java programming language.
- Demonstrate the ability to design, compile, implement, test, and debug simple programs in Java.
- Demonstrate the ability to manipulate numbers and character strings in Java.
- Compare and contrast classes and objects in Java.
- Construct classes through systematic procedures.
- Differentiate between static and non-static methods and variables.
- Demonstrate the ability to program simple and complex decisions in Java.
- Write clearly and concisely about Java programming using proper writing mechanics and technical style conventions.

Grading for this assignment will be based on the following rubric.

Points: 200	**Assignment 1: Financial Portfolio**				
Criteria	**Unacceptable Below 60% F**	**Meets Minimum Expectations 60-69% D**	**Fair 70-79% C**	**Proficient 80-89% B**	**Exemplary 90-100% A**
1. Specifications Weight: 50%	The program does not compile.	The program compiles but does not	The program compiles and executes but	The program compiles, executes, and	The program compiles, executes,

		execute.	produces incorrect results.	produces correct results but does not meet all of the specifications.	produces correct results, and meets all of the specifications.
2. Readability Weight: 10%	The code is not organized and very difficult to read.	The code is poorly organized and difficult to read.	The code is partially organized but readable only by someone who knows the expected end result.	The code is organized and easy to read.	The code is exceptionally organized and very easy to read.
3. Reusability and object-oriented programming constructs Weight: 10%	The code is not organized for reusability.	The code is poorly organized for reusability.	The code is partially organized and some parts of the code could be reused in other programs.	The code is organized and most of the code could be reused in other programs.	The code is exceptionally organized and could be reused as a whole or each routine could be reused.
4. Efficiency Weight: 10%	The code is unnecessarily long and appears to be patched together.	The code is unnecessarily long.	The code is fairly efficient but sacrifices readability and understanding.	The code is efficient without sacrificing readability and understanding.	The code is extremely efficient without sacrificing readability and understanding.
5. Documentation Weight: 10%	No documentation is provided.	The documentation consists of embedded comments but does not help the reader understand the code.	The documentation consists of embedded comments and some header comments separating routines.	The documentation consists of embedded comments and header documentation that is useful in understanding the code.	The documentation consists of embedded comments and clearly explains what the code is accomplishing and how.
6. Include a one (1) page description about your program. Weight: 5%	Did not submit or incompletely included a one (1) page description about your program.	Insufficiently included a one (1) page description about your program.	Partially included a one (1) page description about your program.	Satisfactorily included a one (1) page description about your program.	Thoroughly included a one (1) page description about your program.
7. Clarity, writing mechanics, and formatting requirements Weight: 5%	More than 8 errors present	7-8 errors present	5-6 errors present	3-4 errors present	0-2 errors present

Case Study: Business Applications of Java
Due Week 8 and worth 100 points

Read the article titled, “Business Applications of Java” dated 2014, located in the online course shell.

Write a two to four (2-4) page paper in which you:
1. Summarize the main points of the article.
2. Select two (2) features (e.g., class, object, etc.) of the Java programming language that make it well suited for business applications. Provide one (1) example of a scenario in which each feature is used in business to support your response.
3. Provide one (1) example of a scenario which illustrates that Java programs that make simple and complex decisions are useful in business. Justify your response.
4. Speculate on the overall trend of the application of Java within the business world over the next five (5) years. Justify your response.

Your assignment must follow these formatting requirements:
- Be typed, double spaced, using Times New Roman font (size 12), with one-inch margins on all sides; citations and references must follow APA or school-specific format. Check with your professor for any additional instructions.
- Include a cover page containing the title of the assignment, the student’s name, the professor’s name, the course title, and the date. The cover page and the reference page are not included in the required assignment page length.

The specific course learning outcomes associated with this assignment are:
- Compare and contrast classes and objects in Java.
- Demonstrate the ability to program simple and complex decisions in Java.
- Use technology and information resources to research issues in Java programming.
- Write clearly and concisely about Java programming using proper writing mechanics and technical style conventions.

Grading for this assignment will be based on answer quality, logic / organization of the paper, and language and writing skills, using the following rubric.

Points: 100	**Case Study: Business Applications of Java**				
Criteria	**Unacceptable Below 60% F**	**Meets Minimum Expectations 60-69% D**	**Fair 70-79% C**	**Proficient 80-89% B**	**Exemplary 90-100% A**
1. Summarize the main points of the article. Weight: 10%	Did not submit or incompletely summarized the main points of the article.	Insufficiently summarized the main points of the article.	Partially summarized the main points of the article.	Satisfactorily summarized the main points of the article.	Thoroughly summarized the main points of the article.
2. Select two (2) features (e.g., class, object, etc.) of the Java programming language that make it well suited for business	Did not submit or incompletely selected two (2) features (e.g., class, object, etc.) of the Java	Insufficiently selected two (2) features (e.g., class, object, etc.) of the Java	Partially selected two (2) features (e.g., class, object, etc.) of the Java programming	Satisfactorily selected two (2) features (e.g., class, object, etc.) of the Java	Thoroughly selected two (2) features (e.g., class, object, etc.) of the Java

applications. Provide one (1) example of a scenario in which each feature is used in business to support your response. Weight: 30%	programming language that make it well suited for business applications. Did not submit or incompletely provided one (1) example of a scenario in which each feature is used in business to support your response.	programming language that make it well suited for business applications. Insufficiently provided one (1) example of a scenario in which each feature is used in business to support your response.	language that make it well suited for business applications. Partially provided one (1) example of a scenario in which each feature is used in business to support your response.	programming language that make it well suited for business applications. Satisfactorily provided one (1) example of a scenario in which each feature is used in business to support your response.	programming language that make it well suited for business applications. Thoroughly provided one (1) example of a scenario in which each feature is used in business to support your response.
3. Provide one (1) example of a scenario which illustrates that Java programs that make simple and complex decisions are useful in business. Justify your response. Weight: 30%	Did not submit or incompletely provided one (1) example of a scenario which illustrates that Java programs that make simple and complex decisions are useful in business. Did not submit or incompletely justified your response.	Insufficiently provided one (1) example of a scenario which illustrates that Java programs that make simple and complex decisions are useful in business. Insufficiently justified your response.	Partially provided one (1) example of a scenario which illustrates that Java programs that make simple and complex decisions are useful in business. Partially justified your response.	Satisfactorily provided one (1) example of a scenario which illustrates that Java programs that make simple and complex decisions are useful in business. Satisfactorily justified your response.	Thoroughly provided one (1) example of a scenario which illustrates that Java programs that make simple and complex decisions are useful in business. Thoroughly justified your response.
4. Speculate on the overall trend of the application of Java within the business world over the next five (5) years. Justify your response. Weight: 20%	Did not submit or incompletely speculated on the overall trend of the application of Java within the business world over the next five (5) years. Did not submit or incompletely justified your response.	Insufficiently speculated on the overall trend of the application of Java within the business world over the next five (5) years. Insufficiently justified your response.	Partially speculated on the overall trend of the application of Java within the business world over the next five (5) years. Partially justified your response.	Satisfactorily speculated on the overall trend of the application of Java within the business world over the next five (5) years. Satisfactorily justified your response.	Thoroughly speculated on the overall trend of the application of Java within the business world over the next five (5) years. Thoroughly justified your response.
5. Clarity, writing mechanics, and formatting requirements Weight: 10%	More than 8 errors present	7-8 errors present	5-6 errors present	3-4 errors present	0-2 errors present

Assignment 2: uGrade
Due Week 10 and worth 200 points

This assignment consists of two (2) sections:

- A Java program file
- A screen shot of the output and a description of your Java program

Label each file name according to the section of the assignment for which it is written. Put both sections together in a single zip file and submit the zip file.

Imagine that you work as a Java programmer for a software company. Your company is developing a new interactive learning management tool called uGrade. You have the task of creating the functionality that will allow the user to manually enter names and grades. The program will store this information in parallel arrays. Next, it will prompt the user to sort the date by name or by grade and display the sorted data accordingly in an organized table.

Section 1: Java Program File

1. Create a Java program in which you include:
    - a class called “roster”
    - attributes, which must be last names and numerical grades, and must also be parallel arrays
    - an indefinite loop, which allows the user to manually enter five (5) names and grades and stores them in the array. **Note:** The loop must end either when the user specifies that they are done or when a maximum of five (5) values have been stored in the arrays, i.e., five (5) names and a corresponding grade for each name.
    - an indefinite loop, which prompts the user to select a sorting criterion or to end the program, and must also use bubble sort. **Note:** The user may either select **name** or **grade** as the sorting criteria. The program must use bubble sort to sort that data according to the specific criteria and then use another loop to display the data. This process must continue until the user ends the program.

Section 2: Screen Shot of the Output and Description of Your Java Program

2. Create a screen shot of the output and include a description about your Java program.
    - Submit a screen shot which shows the output of your Java Program. **Note:** Go to http://www.take-a-screenshot.org/ if you need a tutorial on taking a screen shot.
    - Include a one (1) page description about your program. **Note:** Use MS Word for your program description, and place the screen shot of the output from your Java program into the Word file as an attached image.

    The output should look like this if the user chose to sort by name:

```
Name          Numerical Grade
Bailey        97
David         88
Ericson       79
Frank         99
Manning       91
```

The output should look like this if the user chose to sort by grade:

```
Name         Numerical Grade
Frank        99
Bailey       97
Manning      91
David        88
Ericson      79
```

Note: The name and grade pairs must stay together.

Section 1 and Section 2 will be graded based on the following:

1. The program must compile, execute, produce correct results, and meet all of the specifications stated in Section 1.

Additionally you must:

2. Organize the code for user readability.
3. Organize the code for reusability.
4. Organize the code for efficiency.
5. Provide documentation with embedded comments for reader understanding.
6. Include a one (1) page description about your program.

The specific course learning outcomes associated with this assignment are:

- Demonstrate the proper use and application of syntax in the Java programming language.
- Demonstrate the ability to design, compile, implement, test, and debug simple programs in Java.
- Demonstrate the ability to manipulate numbers and character strings in Java.
- Compare and contrast classes and objects in Java.
- Construct classes through systematic procedures.
- Discuss object-oriented design principles.
- Compare and contrast abstract and concrete data types.
- Demonstrate the ability to implement generic classes and methods.
- Declare and use interface types.
- Demonstrate the ability to program simple and complex decisions in Java.
- Implement loops for repetitive tasks.
- Compare and contrast definite loops and indefinite loops.
- Compare and contrast arrays and array lists in Java.
- Write clearly and concisely about Java programming using proper writing mechanics and technical style conventions.

Grading for this assignment will be based on the following rubric.

Points: 200	**Assignment 2: uGrade**				
Criteria	**Unacceptable Below 60% F**	**Meets Minimum Expectations 60-69% D**	**Fair 70-79% C**	**Proficient 80-89% B**	**Exemplary 90-100% A**
1. Specifications Weight: 50%	The program does not compile.	The program compiles but does not execute.	The program compiles and executes but produces incorrect results.	The program compiles, executes, and produces correct results but does not meet all of the specifications.	The program compiles, executes, produces correct results, and meets all of the specifications.
2. Readability Weight: 10%	The code is not organized and very difficult to read.	The code is poorly organized and difficult to read.	The code is partially organized but readable only by someone who knows the expected end result.	The code is organized and easy to read.	The code is exceptionally organized and very easy to read.
3. Reusability and object-oriented programming constructs Weight: 10%	The code is not organized for reusability.	The code is poorly organized for reusability.	The code is partially organized and some parts of the code could be reused in other programs.	The code is organized and most of the code could be reused in other programs.	The code is exceptionally organized and could be reused as a whole or each routine could be reused.
4. Efficiency Weight: 10%	The code is unnecessarily long and appears to be patched together.	The code is unnecessarily long.	The code is fairly efficient but sacrifices readability and understanding.	The code is efficient without sacrificing readability and understanding.	The code is extremely efficient without sacrificing readability and understanding.
5. Documentation Weight: 10%	No documentation is provided.	The documentation consists of embedded comments but does not help the reader understand the code.	The documentation consists of embedded comments and some header comments separating routines.	The documentation consists of embedded comments and header documentation that is useful in understanding the code.	The documentation consists of embedded comments and clearly explains what the code is accomplishing and how.
6. Include a one (1) page description about your program. Weight: 5%	Did not submit or incompletely included a one (1) page description about your program.	Insufficiently included a one (1) page description about your program.	Partially included a one (1) page description about your program.	Satisfactorily included a one (1) page description about your program.	Thoroughly included a one (1) page description about your program.

7. Clarity, writing mechanics, and formatting requirements Weight: 5%	More than 8 errors present	7-8 errors present	5-6 errors present	3-4 errors present	0-2 errors present

Weekly Course Schedule

The purpose of the course schedule is to give you, at a glance, the required preparation, activities, and evaluation components of your course. For more information about your course, whether on-ground or online, access your online course shell.

The expectations for a 4.5 credit hour course are for students to spend 13.5 hours in weekly work. This time estimate includes preparation, activities, and evaluation regardless of the delivery mode.

Instructional Materials

In order to be fully prepared, obtain a copy of the required textbooks and other instructional materials prior to the first day of class. When available, Strayer University provides a link to the first three (3) chapters of your textbook(s) in eBook format. Check your online course shell for availability.

Review the online course shell or check with your professor to determine whether Internet-based assignments and activities are used in this course.

Strayer students are encouraged to purchase their course materials through the Strayer Bookstore. http://www.strayerbookstore.com If a lab is required for the course, the Strayer Bookstore is the only vendor that sells the correct registration code so that Strayer students may access labs successfully.

Software

The Java Platform and an Integrated Development Environment (IDE) is required for this course.

- Download the Java Development Kit (JDK) from http://www.oracle.com/technetwork/java/javase/downloads/index.html.
- Download the Eclipse IDE from http://www.eclipse.org/downloads/. Most users will want the Eclipse IDE for Java Developers, Windows 32 bit.

Students may use other Java IDEs to develop the code for their assignments and labs as long as the code and assignments can be replicated for the instructor's review. A suggested alternative to Eclipse that some consider less powerful but easier to use is JGrasp. JGrasp can be downloaded for Windows and Mac from http://spider.eng.auburn.edu/user-cgi/grasp/grasp.pl?;dl=download_jgrasp.html.

Kaltura Video Sessions

Instructors will record video or screen capture sessions using **Kaltura** in Weeks 2 and 4. Your instructor may also provide these sessions on a weekly basis to act as student tutorials on the following applicable procedures:

- Running and navigating the software or labs that will be used in this course
- Demonstrating the use of relevant programming languages and tools through screen capture and live navigation
- Logging into the lab environments and running a sample lab required in this course (if labs are required)
- Demonstrating the downloading and installing of software to a student's computer (if necessary) or using the software already installed in the Strayer University campus labs
- Locating or finding files and other student materials that may be required to use in assignments
- Submitting assignments in Blackboard

You are strongly advised to use the material provided by faculty in these sessions. Online and Ground instructors will post the Kaltura video recorded sessions to the Instructor Insights folder within the weekly tabs of the online course shell (Blackboard).

Discussions

To earn full credit in an online threaded discussion, students must have one original post and a minimum of one other post per discussion thread.

Please note: Material in the online class will be made available three weeks at a time to allow students to work ahead, however, faculty will be focused on and responding only to the current calendar week. As it is always possible that students could lose their work due to unforeseen circumstances, it is a best practice to routinely save a working draft in a separate file before posting in the course discussion area.

Professors hold discussions during class time for on-ground students. Check with your professor if any additional discussion participation is required in the online course shell outside of class hours.

Tests

Tests (quizzes, midterm and final exams, essay exams, lab tests, etc.) are available for student access and completion through the online course shell. Check the online course shell to determine how students are expected to take the tests. Do not change these questions or their point values in any way. This disrupts the automated grade book preset in the online course shell.

- Online students are to complete the test by Monday 9:00 a.m. Details regarding due dates are posted in the Blackboard Calendar tool.
- On-ground students are to complete the tests after the material is covered and before the next class session.

Assignments

A standardized performance grading rubric is a tool your professor will use to evaluate your written assignments. Review the rubric before submitting assignments that have grading rubrics associated with them to ensure you have met the performance criteria stated on the rubric.

Grades are based on individual effort. There is no group grading; however, working in groups in the online or on-ground classroom is acceptable.

Assignments for online students are always submitted through the online course shell. On-ground professors will inform students on how to submit assignments, whether in paper format or through the online course shell.

Association for Computing Machinery (ACM) Digital Library

The ACM Digital Library is a complete collection of all of ACM's publications, including ACM journals, conference proceedings, magazines, newsletters, and multimedia titles. The ACM Digital Library contains the largest and most complete full-text archive of articles on computing available today, consisting of: 2.0+ million pages of full-text articles, 20,000 new full-text articles added each year, 40+ high-impact journals, 270+ conference proceedings titles, 9 magazines (including the flagship Communications of the ACM), and 43 special interest groups contributing content.

You are encouraged to search the ACM Digital Library for full-text articles for your writing assignments and term papers refereed referenced material. For more information on the ACM Digital Library, please watch the video located in the Student Center tab of the online course shell (Blackboard).

To access the ACM Digital Library:

Students:

1. Login to iCampus: http://icampus.strayer.edu
2. Click on Campus & Library
3. Mouse-over or click on Learning Resource Center
4. Click on Databases

5. Scroll down to Information Systems / Computing
6. Click on ACM Digital Library

Resources

The Resource Center navigation button in the online course shell contains helpful links. Strayer University Library Resources are available here as well as other important information. You should review this area to find resources and answers to common questions.

Technical support is available for the following:

- For **technical questions**, please contact Strayer Online Technical Support by logging in to your iCampus account at https://icampus.strayer.edu/login and submitting a case under "Student Center," then "Submit Help Ticket." If you are unable to log in to your iCampus account, please contact Technical Support via phone at (877) 642-2999.

- For **concerns with your class**, please access the Solution Center by logging in to your iCampus account at https://icampus.strayer.edu/login and submitting a case under "Student Center," then "Submit Help Ticket." If you are unable to log in to your iCampus account, please contact the IT Help Desk at (866) 610-8123 or at mailto:IThelpdesk@Strayer.edu.

TurnItIn.com is an optional online tool to assess the originality of student written work. Check with your professor for access and use instructions.

The **Strayer Policies** link on the navigation bar in the online course shell contains academic policies. It is important that students be aware of these policies.