# Washington Township School District

***The mission of the Washington Township Public Schools*** *is to provide a safe, positive, and progressive educational environment that provides opportunity for all students to attain the knowledge and skills specified in the NJ Learning Standards at all grade levels, so as to ensure their full participation in an ever-changing world as responsible, self-directed, and civic-minded citizens.*

| | |
|---|---|
| **Course Title:** | **Intro to Computer Science - JAVA** |
| **Grade Level(s):** | 9-12 |

| | | | | | | |
|---|---|---|---|---|---|---|
| **Duration:** | *Full Year:* | **X** | *Semester:* | | *Marking Period:* | |

**Course Description:**

This full year, elective course is designed to introduce students to the concepts of Computer Science using the JAVA programming language and is intended as a prerequisite to the "AP Computer Science - JAVA" course. This course is suggested for those students who have successfully completed grade 8 Algebra 1 with an 85 or better. This course may also be taken concurrently with Geometry A or after completion of Geometry A with an 85 or better.

To succeed, students must be willing and able to work individually, participate in classroom discussions, complete assignments in a timely fashion, be capable of logical thinking, be able to break down problems into simple, sequential tasks, and be detail oriented.

During the course of the year, the students will be introduced to and become proficient with the history of computing & ethics, computer components, basic web page design, using an IDE (Integrated Development Environment), understanding, and using the JAVA language and its syntax, the use of the sequence, decision, and loop structures and ultimately Object-Oriented Programming (OOP) design principles. Emphasis will be placed on algorithm development, program structure, documentation, language syntax, and problem-solving skills. The students will create algorithms and create original programs that meet assignment criteria and will take quizzes and tests to demonstrate their mastery of the content. Classwork includes lectures, class discussions, creating a JAVA Quick Reference as a reference tool for Java syntax & structure, program design, writing programs, and testing / debugging / revising programs to meet the assigned design criteria. Homework includes reading assignments, answering questions at the end of each unit, creating program algorithms, designing programs / classes / methods, review exercises and studying for quizzes and tests.

**Grading Procedures:**

The final grade will be a composite of homework, classwork, quiz, and test scores. Individual program design / completeness / adherence to design specifications, and the student's mastery of the areas listed above will be assessed. The student will pass the course with a minimum overall average of 70%. The specific grading system will be explained to the students by the individual teacher.

**Primary Resources:**

JAVA – An Introduction to Problem Solving & Programming

# Washington Township Principles for Effective Teaching and Learning

- Implementing a standards-based curriculum
- Facilitating a learner-centered environment
- Using academic target language and providing comprehensible instruction
- Adapting and using age-appropriate authentic materials
- Providing performance-based assessment experiences
- Infusing 21st century skills for College and Career Readiness in a global society

| | |
|---|---|
| **Designed by:** | **William Faust** |

| | |
|---|---|
| **Under the Direction of:** | **Carole English** |

**Written:** July 2022

**Revised:** _____

**BOE Approval:** _____

**Unit Title: 1) History of Computing and Computer Components**

**Unit Description:**

This unit begins by examining the "History of Computers". The development of computing spans from precomputing – with items such as the abacus – through to today's modern computer. Significant people, technologies, and achievements are presented and discussed.

The history of the Internet, networking, ethics in Computer Science, and data security is also discussed. A simple web page project will be assigned to develop the student's ability to create a web page utilizing HTML code.

The unit concludes with a discussion of the basic hardware / software components of a modern computer and in what ways they are relevant to a programmer. The global impact of advancements in computer science is also examined.

**Unit Duration:** 6 days

## Desired Results

**Standard(s):**

8.1 Computer Science
8.2 Design Thinking

**Indicators:**

8.1.12.CS.1: Describe ways in which integrated systems hide underlying implementation details to simplify user experiences.
8.1.12.CS.3: Compare the functions of application software, system software, and hardware.
8.1.12.NI.1: Evaluate the scalability and reliability of networks, by describing the relationship between routers, switches, servers, topology, and addressing.
8.1.12.NI.2: Evaluate security measures to address various common security threats.
8.1.12.NI.3: Explain how the needs of users and the sensitivity of data determine the level of security implemented.
8.1.12.IC.1: Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.
8.1.12.IC.3: Predict the potential impacts and implications of emerging technologies on larger social, economic, and political structures, using evidence from credible sources.
8.1.12.DA.2: Describe the trade-offs in how and where data is organized and stored.
8.1.12.DA.3: Translate between decimal numbers and binary numbers.
8.1.12.DA.4: Explain the relationship between binary numbers and the storage and use of data in a computing device.

8.2.12.ED.5: Evaluate the effectiveness of a product or system based on factors that are related to its requirements, specifications, and constraints (e.g., safety, reliability, economic considerations, quality control, environmental concerns, manufacturability, maintenance and repair, ergonomics).
8.2.12.ED.6: Analyze the effects of changing resources when designing a specific product or system (e.g., materials, energy, tools, capital, labor).

8.2.12.ITH.1: Analyze a product to determine the impact that economic, political, social, and/or cultural factors have had on its design, including its design constraints.
8.2.12.ITH.3: Analyze the impact that globalization, social media, and access to open-source technologies has had on innovation and on a society's economy, politics, and culture.
8.2.12.NT.1: Explain how different groups can contribute to the overall design of a product.
8.2.12.EC.1: Analyze controversial technological issues and determine the degree to which individuals, businesses, and governments have an ethical role in decisions that are made.
8.2.12.EC.2: Assess the positive and negative impacts of emerging technologies on developing countries and evaluate how individuals, non-profit organizations, and governments have responded.
8.2.12.EC.3: Synthesize data, analyze trends, and draw conclusions regarding the effect of a technology on the individual, culture, society, and environment and share this information with the appropriate audience.
8.2.12.ETW.4: Research historical tensions between environmental and economic considerations as driven by human needs and wants in the development of a technological product and present the competing viewpoints

**Understandings:**
*Students will …*

Identify & discuss the significant people, machines, developments, and technologies throughout the history of the computer. (Generations 1-5, & History of the Internet & Cloud computing, Ethics in Computer Science, and Data Security)

Be able to identify and briefly describe the major components of a typical PC computer system
1. CPU
2. Motherboard
3. Storage (Hard drives, SSD drives, Flash drives, etc.)
4. Monitor
5. Peripherals

Create a simple web page utilizing html coding.

**Essential Questions:**

1. What can we learn from examining the history of computers?
2. Identify the famous people related to the history of computing and explain their impact on the history of the development of computers.
3. Identify the famous machines throughout the history of computing and explain their impact on the history of the development of computers.
4. Outline the Code of Conduct for Computer Ethics.
5. What measures can be taken to secure your personal digital data?
6. What are the components of a modern computer and why are they relevant to a computer programmer?
7. What are the major subcomponents of a modern computer and explain their function?
8. What is the process for creating a simple web page with an embedded applet?
9. What are some of the ramifications of the advancement of computer science on society's future?

## Assessment Evidence

**Performance Tasks:**

Web Page project

**Other Evidence:**

**Formative Assessments:**
Unit Quizzes (2)

**Summative Assessment(s)**
Unit Test

**Benchmarks:**

Web Page Project

## Learning Plan

**Learning Activities:**

Complete guided notes on History of Computers (using PowerPoint Presentation).
- "Pre-Computers" (Before 1940) – Hardware Software, Special Machines & People
- First Generation (1940 to 1954) – Hardware Software, Special Machines & People
- Second Generation (1954 to 1964) – Hardware Software, Special Machines & People
- Third Generation (1964 to 1970) – Hardware Software, Special Machines & People
- Fourth Generation (1971 to 1980) – Hardware Software, Special Machines & People
- Fifth Generation (1980 and beyond) – Hardware Software, Special Machines & People

View artifacts from computer antiquity.

Complete guided notes, participate in discussion of:
- History if the Internet Networking (ARPANET, Packet Switching, FTP, Telnet, Gopher/Archie, HTTP, etc.),
- Ethics (Piracy, Copyright Law, "Student Acceptable Use of Computer Network/ Computers and Resources and Data Security" school document,
- Data Security (Personal Data Security Best Practices, Password Strength).

Complete guided notes on modern computer components and their relevance/application to programming:
- Motherboard, CPU, Cache, Architecture, Memory, Expansion Slots, Ports, Storage, Monitors, Peripherals, etc. (Text 1.1)

Dissect a computer into its component parts and identify them/describe their purpose.

Create a basic web page using html tags. (www.w3schools.com website for web page creation)

**Resources:**

JAVA – An Introduction to Problem Solving & Programming: Chapter 1.1
Guided Notes packet (electronic)
PowerPoint presentation on History of Computing
Student Computers with Network Drive Access & Internet Access

| Unit Modifications for Special Population Students | |
|---|---|
| **Advanced Learners** | Ask reflective and extension questions to build on classroom knowledge to develop a deeper understanding.<br>Opportunities for extended learning through additional programs (see text for possibilities).<br>Pose "What if…" questions. |
| **Struggling Learners** | Redirect student to use Quick Reference.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher.<br>Redirect student attention.<br>After school availability for help.<br>Internet resources (videos on topic, websites relevant to the particular topic, etc.). |

| | |
|---|---|
| **English Language Learners** | Use a translator device.<br>Provide access to language dictionary, instructor, or any other means to help interpret any language/communication difficulties.<br>Rephrase questions for student clarification.<br>Have student create vocabulary flash cards in addition to Quick Reference. |
| **Special Needs Learners** | Follow IEP modifications.<br>Redirect student attention.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher. |
| **Learners with a 504** | Refer to page four in the Parent and Educator Resource Guide to Section 504 to assist in the development of appropriate plans. |

## Interdisciplinary Connections

**Indicators:**
**ELA: RST.9-10.4; RST.11-12.4; WHST.9-10.2.D**
**Science: ETS1.A: Defining and Delimiting Engineering Problems; ETS1.B: Developing Possible Solutions; ETS1.C: Optimizing the Design Solution**
**Social Studies: 6.1.12.EconNE.3.a, 6.1.12.EconNE.16a, 6.1.12.EconNE.16b**

## Integration of 21st Century Skills

**Indicators:**

*From the Partnership for 21st Century Skills (P21), the deeper learning competencies and skills for 21st century learning in this unit include:*

Communication
Critical thinking
Creativity

**Unit Title: 2) Windows OS and the Program Development Cycle (Optional: Computer Logic & Binary)**

**Unit Description:**

This unit begins by familiarizing the student with the Windows Operating System (OS); how to navigate it, the standard features of any Windows program, the file system and how to create and manipulate files / folders in the OS.

The unit concludes with the very beginnings of the learning how to program a computer. The Program Development Cycle is introduced and established as a framework for all future program creation. The students are introduced to the Integrated Development Environment (IDE) which is the software used to create their programs. Two small programs are created during the introduction. Students are then presented with some of the basic guidelines regarding JAVA syntax and programming conventions in general. Programming errors are also briefly introduced.

Time Permitting, this unit also introduces the programming student to the basics of Computer Logic (binary, logic gates, and circuits), the Binary Numbering System, and how to convert and perform addition in Binary.

**Unit Duration:** 5 days

## Desired Results

**Standard(s):**

8.1 Computer Science
8.2 Design Thinking

**Indicators:**

8.1.12.CS.2: Model interactions between application software, system software, and hardware.
8.1.12.CS.3: Compare the functions of application software, system software, and hardware.
8.1.12.AP.5: Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
8.1.12.AP.6: Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.

8.2.12.ED.5: Evaluate the effectiveness of a product or system based on factors that are related to its requirements, specifications, and constraints (e.g., safety, reliability, economic considerations, quality control, environmental concerns, manufacturability, maintenance and repair, ergonomics).

Optional:
8.1.12.DA.3: Translate between decimal numbers and binary numbers.
8.1.12.DA.4: Explain the relationship between binary numbers and the storage and use of data in a computing device.

**Understandings:**
*Students will …*

Navigate & use the basics of the Windows Operating System (OS).
Create a text document using Notepad.
Navigate, copy & delete folders & files using Windows File Explorer.

Define and recite the Program Development Cycle
- Define the problem
- Design Algorithm
- Create Interface
- Code
- Test/Debug
- Documentation

Create an application using NetBeans.
Create a simple graphics application using NetBeans.
Apply comments in a JAVA program.
Define, write & use valid identifiers (key words & variable names in the JAVA language).
Define & use proper syntax when writing a JAVA program.
Compare & contrast compiled vs. interpreted programming languages.
Identify & describe the 3 different types of programming errors: Syntax, Logical, Run-Time.

Optional:
Define & Describe the AND, OR, NOT, XOR Logic Gates.
Follow the circuit diagram of a Half Adder Circuit.
Define & describe the Binary number system & compare to the Decimal system.
Convert from Binary to Decimal & vice versa.
Add in Binary.

**Essential Questions:**

1. What are the names of the components of the Windows OS desktop?
2. Explain the process of creating a file? Deleting it? Creating a folder? Moving a file?
3. Differentiate between a List & a ComboBox?
4. When action occurs when clicking on a button (when press down, when release, or both)?
5. Explain the process by which a program is created?
6. Programs can be millions of lines of code, where do you start?
7. Outline the process that is the most efficient way of developing and releasing a working program?
8. What are the rules regarding the use of the JAVA programming language and programming in general (…how do I write code)?
9. Organize the rules pertaining to legal *identifiers* (key words & variable names, rules regarding capitalization, indentation, & symbology in JAVA).

Optional:
10. What is Computer Logic and how does it function?
11. Draw and explain the function of an AND gate, and an OR gate.
12. What does a “half adder” circuit look like, what does it do, and how does it do it?
13. What is the Binary Numbering System and how do we perform conversions and addition operations with it?
14. Explain the process to convert form Binary to Decimal? Vice Versa?
15. Explain the process for adding in the Binary Numbering System?

## Assessment Evidence

**Performance Tasks:**

Navigate, copy & delete folders & files using Windows File Explorer

**Other Evidence:**

**Formative Assessments:**
Unit Quizzes (3)

**Summative Assessment(s)**
Unit Test

**Benchmarks:**

Unit Test

## Learning Plan

**Learning Activities:**

Students will navigate through the Windows OS with teacher guidance, identifying each component of the OS and/or running program:
Desktop, Taskbar, System Tray, Title Bar, Menu, Borders, Ribbon and interaction/manipulation of each of these items.
Student will perform all directed file operations (creating text file, folder creation, moving, deleting, etc.) using Notepad and/or File Explorer.

Complete guided notes on the Programming Development Cycle: (Text 1.1, 3)
6 Steps of detailed explanation of each, how to apply when writing a program.
Create an Application (HelloWorld) & a simple graphics application (HelloWorldSGA) using the NetBeans IDE with teacher guidance (demo of the IDE - it's purpose, capabilities, and process of creating a program). (Text 1.2)
Follow examples presented in class.
Update Quick Reference as necessary.
Complete assigned homework.

Optional:
Complete guided notes on Binary Numbering System:
Logic Gates, Binary Numbering System, Conversion (Decimal to Binary and Vice Versa), Binary Addition.
Follow examples presented in class (in text & on supplemental handouts).
Complete assigned homework.

**Resources:**

JAVA – An Introduction to Problem Solving & Programming: Chapter 1.1-1.3
Guided Notes packet
Student Computers with NetBeans IDE, Network Drive Access & Internet Access

| Unit Modifications for Special Population Students | |
|---|---|
| **Advanced Learners** | Ask reflective and extension questions to build on classroom knowledge to develop a deeper understanding.<br>Opportunities for extended learning<br><br>Additional Programs (see text for possibilities).<br>Pose "What if…" questions. |
| **Struggling Learners** | Redirect student to use Topical Index.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher.<br>Redirect student attention.<br>After school availability for help.<br>Internet resources (videos on topic, websites relevant to the particular topic, etc.). |
| **English Language Learners** | Use a translator device.<br>Provide access to language dictionary, instructor, or any other means to help interpret any language/communication difficulties.<br>Rephrase questions for student clarification.<br>Have student create vocabulary flash cards in addition to Topical index. |
| **Special Needs Learners** | Follow IEP modifications.<br>Redirect student attention.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher. |
| **Learners with a 504** | Refer to page four in the Parent and Educator Resource Guide to Section 504 to assist in the development of appropriate plans. |

## Interdisciplinary Connections

**Indicators:**

**ELA: RST.9-10.4; RST.11-12.4**
**Mathematics: M1, 2, 4, 5, 7, 8; N-Q**

## Integration of 21st Century Skills

**Indicators:**

*From the Partnership for 21st Century Skills (P21), the deeper learning competencies and skills for 21st century learning in this unit include:*

Communication
Critical thinking
Creativity

**Unit Title: 3) Objects, Classes & Variables**

**Unit Description:**

This unit introduces the students into the concept of Objects & Object-Oriented Programming (OOP). How Objects are represented by Classes and the anatomy of a class (variables & methods) are studied. Some of the standard JAVA class methods are then further explored.

The concept of variables is introduced next. Variables are categorized into two basic types in JAVA: ***primitive*** types and ***object*** types.

The student is first introduced to the primitive variable type. How to name, declare, and initialize primitive variables is studied. The concept of Constants and their essential use are also introduced. The student is then taught how to manipulate primitive variables through the relational operators (add, subtract, multiply, integer and floating-point division, modulo division, etc.) and how conversion and casting operates.

Finally, the student is taught about object type variables. How to name, import, declare, and instantiate object variables is studied. The passing of parameters to object variables is introduced. The Scanner Class is utilized to create a scanner object to facilitate keyboard input to the student's programs.

**Unit Duration:** 20 days

## Desired Results

**Standard(s):**

8.1 Computer Science
8.2 Design Thinking

**Indicators:**

8.1.12.CS.2: Model interactions between application software, system software, and hardware.
8.1.12.CS.3: Compare the functions of application software, system software, and hardware.
8.1.12.DA.2: Describe the trade-offs in how and where data is organized and stored.
8.1.12.AP.1: Design algorithms to solve computational problems using a combination of original and existing algorithms.
8.1.12.AP.5: Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
8.1.12.AP.6: Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
8.1.12.AP.8: Evaluate and refine computational artifacts to make them more usable and accessible.
8.1.12.AP.9: Collaboratively document and present design decisions in the development of complex programs.

8.2.12.ED.1: Use research to design and create a product or system that addresses a problem and make modifications based on input from potential consumers.
8.2.12.ED.5: Evaluate the effectiveness of a product or system based on factors that are related to its requirements, specifications, and constraints (e.g., safety, reliability, economic considerations, quality control, environmental concerns, manufacturability, maintenance and repair, ergonomics).

**Understandings:**

*Students will …*

Define what a JAVA Object is along with its associated attributes & methods.
Relate how an object is defined by a Class.
Define, describe & give examples of Encapsulation & Inheritance.
Use the print( ) & println( ) methods in a JAVA program.
Define, compare & contrast, Characters & Strings in the JAVA programming language.
List & use escape sequences in a JAVA program.

Define & name variables.
List the 4 types of primitive variables along with their defining characteristics.
Declare variables.
Define, name & declare constants.
List & describe the arithmetic operators in JAVA.
Define the Assignment operator.
Create arithmetic expressions using variables, arithmetic operators & the assignment operator.
Define & describe Narrowing & Widening conversions.
List & describe the 3 ways conversions can occur.
Be able to cast variables into other object types.

Define the *new* operator, constructors, & parameters.
Create instances of objects using the new operator in conjunction w/ class's constructor.
Access object's methods utilizing the "dot operator".
Pass parameters to constructors to properly instantiate an object.
Use Import Statements to access classes not in the JAVA Standard Library.
Use the Scanner class to create Scanner objects and obtain input from the user.

**Essential Questions:**

1. Explain why OOP was developed?
2. Differentiate between a Class and an Object? How is an Object represented by a Class?
3. Explain how can you join multiple pieces of text and calculated values together to produce meaningful output?
4. Justify which of the 4 types of primitive variables in JAVA you would use in each situation?
    a. A person's weight
    b. Number of busses needed for a run
    c. A person's name
    d. Determining if the lights are on or off
5. What is the difference between variable declaration and initialization / instantiation?
6. Explain the function of the assignment operator?
7. What are the three types of division and relate each to a real-life situation?
8. Explain the 3 ways data conversions can occur?
9. Explain the process of creating **Object** variable and relate that to that of creating a primitive type?
10. Explain what a constructor is, how do you run one, and pass information to it?
11. Outline the process by which you access the methods provided by a Class?
12. What is the syntax for getting keyboard input from the user (numeric and string)?
13. Write a program that would allow the user to enter numeric data, manipulate it algebraically, and display the results on the screen.

## Assessment Evidence

**Performance Tasks:**

Unit Programs (2):

Coordinates Program,
Temperature Converter II
(Or equivalents)

**Other Evidence:**

**Formative Assessments:**
Unit Quizzes (2)

**Summative Assessment(s)**
Unit Test

**Benchmarks:**

Coordinates Program

## Learning Plan

**Learning Activities:**

Complete guided notes on Objects and Classes: (Text 1.3)
- Attributes, Methods, Classes (template), Object (created from "template"),
- Encapsulation, Inheritance (brief description),
- Methods used for console printing (syntax & examples),
- Brief description of characters vs. Strings, and then escape sequences.

Enter the print( ) & println( ) methods and escape sequences into the Quick Reference.
Follow examples presented in class (in text & on supplemental handouts).
Complete assigned homework.

Complete guided notes on Variables (***primitive*** type): (Text 2.1, 4)
- Need for, the 4 Types, how to declare & initialize them,
- Constants, how to declare & initialize them,
- Arithmetic operations (including modulo),
- Assignment Operator and its use with above content,
- Data Conversion (Narrowing & Widening, Assignment/Arithmetic/Casting conversions).

Follow examples presented in class (in text & in notes).
Copy & run the sample program (Temp Converter) from notes as practice.
Complete assigned homework.
Write JAVA program(s) utilizing the topics outlined here within: Coords (or equivalent).

Complete guided notes on Variables (***object*** type): (Text 2.3)
- Compare & contrast with primitive types, instantiation vs. initialization, running constructors, variable contents (value vs. memory reference).
- Accessing methods using the dot operator.
- Using returned values from methods and passing values to them.
- Imports.
- Scanner class (keyboard input) and it's relevant methods.

Follow examples presented in class (in text & in notes).
Enter the Scanner Object & its methods into the Quick Reference.
Copy & run the two sample programs from notes as practice.
Complete assigned homework.
Write JAVA program(s) utilizing the topics outlined here within: Temp Converter II (or equivalent)

**Resources:**

JAVA – An Introduction to Problem Solving & Programming: Chapter 1.3, 2.1, 2.3-2.4
Guided Notes packet (electronic)
Student Computers with NetBeans IDE, Network Drive Access & Internet Access

## Unit Modifications for Special Population Students

| | |
|---|---|
| **Advanced Learners** | Ask reflective and extension questions to build on classroom knowledge to develop a deeper understanding.<br>Opportunities for extended learning through additional programs (see text for possibilities).<br>Pose "What if…" questions. |
| **Struggling Learners** | Redirect student to use Quick Reference.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher.<br>Redirect student attention.<br>After school availability for help.<br>Internet resources (videos on topic, websites relevant to the particular topic, etc.). |
| **English Language Learners** | Use a translator device.<br>Provide access to language dictionary, instructor, or any other means to help interpret any language/communication difficulties.<br>Rephrase questions for student clarification.<br>Have student create vocabulary flash cards in addition to Quick Reference. |
| **Special Needs Learners** | Follow IEP modifications.<br>Redirect student attention.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher. |
| **Learners with a 504** | Refer to page four in the Parent and Educator Resource Guide to Section 504 to assist in the development of appropriate plans. |

## Interdisciplinary Connections

**Indicators:**

**ELA: RST.9-10.4,7; RST.11-12.4,7; WHST.9-10.2.D**
**Mathematics: M1, 2, 4, 5, 6; N-Q.2-3, A-SSE.1, A-CED.2, A-APR.6, F-LE.5**
**Science: ETS1.A: Defining and Delimiting Engineering Problems; ETS1.B: Developing Possible Solutions; ETS1.C: Optimizing the Design Solution**

## Integration of 21st Century Skills

**Indicators:**

*From the Partnership for 21st Century Skills (P21), the deeper learning competencies and skills for 21st century learning in this unit include:*

Collaboration
Communication
Critical thinking
Creativity

**Unit Title: 4) Formatting, GUI Applications, the String Class & its methods**

**Unit Description:**

This unit begins by enhancing the students' capabilities with a quick discussion of JAVA's formatting capabilities within the print( ) methods. Students will learn to format their output to a specified decimal place.

The unit continues with the knowledge necessary to create GUI applications (full Windows applications verses simple black & white screen "console" applications as previous programs were). Topics include Labels, Buttons, TextFields, Check/Combo/ListBoxes, Panels, Group Buttons, showing and disposing of forms, along with the associated properties for each component previously named.

Finally, the students are provided with an in-depth examination of the String class with all its capabilities (searching, extracting sub-strings, length, comparing, etc.)

Optional:
A Simple Graphics Application (**non**-GUI) template will be provided to the students providing them the ability to create purely graphical output, e.g., lines, rectangles, ovals, etc. and text in color.

**Unit Duration:** 14 days

## Desired Results

**Standard(s):**

8.1 Computer Science
8.2 Design Thinking

**Indicators:**

8.1.12.CS.1: Describe ways in which integrated systems hide underlying implementation details to simplify user experiences.
8.1.12.CS.2: Model interactions between application software, system software, and hardware.
8.1.12.CS.3: Compare the functions of application software, system software, and hardware.
8.1.12.CS.4: Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors.
8.1.12.DA.1: Create interactive data visualizations using software tools to help others better understand real world phenomena, including climate change.
8.1.12.DA.2: Describe the trade-offs in how and where data is organized and stored.
8.1.12.DA.6: Create and refine computational models to better represent the relationships among different elements of data collected from a phenomenon or process.
8.1.12.AP.1: Design algorithms to solve computational problems using a combination of original and existing algorithms.
8.1.12.AP.2: Create generalized computational solutions using collections instead of repeatedly using simple variables.
8.1.12.AP.5: Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
8.1.12.AP.6: Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
8.1.12.AP.8: Evaluate and refine computational artifacts to make them more usable and accessible.

8.1.12.AP.9: Collaboratively document and present design decisions in the development of complex programs.

8.2.12.ED.1: Use research to design and create a product or system that addresses a problem and make modifications based on input from potential consumers.
8.2.12.ED.5: Evaluate the effectiveness of a product or system based on factors that are related to its requirements, specifications, and constraints (e.g., safety, reliability, economic considerations, quality control, environmental concerns, manufacturability, maintenance and repair, ergonomics).
8.2.12.ITH.2: Propose an innovation to meet future demands supported by an analysis of the potential costs, benefits, trade-offs, and risks related to the use of the innovation.

**Understandings:**
*Students will …*

Describe the use of the formatting specifiers available in the print( ) methods.
Use formatting specifiers to produce formatted output.

Define GUI.
Create GUI forms.
Manipulate GUI forms & controls by using the following properties:
Title, Text, Foreground, Border, Horizontal Alignment, Icon, setVisible, LocationByPlatform, defaultCloseOperation, HorizontalTextPosition, HorizontalAlignment, & Model properties
Create GUI forms by using the following controls:

- Labels
- Buttons
- Text Fields
- Image Control
- Check, Combo & List Boxes
- Panels & GroupButton component

Show and dispose of forms.

Declare & initialize strings (including "null" strings).
Use the String methods to manipulate string objects.

- compareTo( )
- equals( )
- indexOf( ) (both of them)
- length( )
- substring( ) (both of them)
- toLowerCase( ) & toUpperCase( )

**Essential Questions:**

1. What code would you write if a programming situation required you to format your output to 2 decimal places?
2. How do you design a form using NetBeans Form Designer?
3. All Windows programs are "____ driven" *(answer…EVENT driven)* programs…explain what this means.
4. Where do you write code so that will execute when a user clicks on a button?
5. What method(s) would allow you to search within a string for certain characters? What if those characters appear more than once?
6. Write a program that would search for the individual components of a user's entered personal information and display them in a GUI form.
7. Optional: Write as Simple Graphics Application that would display a Snowman.

## Assessment Evidence

**Performance Tasks:**

Unit Programs (5):
Length-O-Skid,
My 1st GUI, My 2nd GUI
Address Parser

**Other Evidence:**

**Formative Assessments:**
Unit Quiz (1)

**Summative Assessment(s)**

| | |
|---|---|
| Snowman<br>(Or equivalents) | Unit Test |

**Benchmarks:**

Length-O-Skid program
Address Parser program

## Learning Plan

**Learning Activities:**

Complete guided notes on Formatting Output (printf( ) method. (Text 2.3)
Enter the formatting specifiers for the printf( ) method into the Quick Reference.
Write JAVA program(s) utilizing the topics outlined here within: Length-O-Skid (or equivalent)

Create a Windows GUI application with teacher guidance (My 1st GUI).
Enter all GUI related properties into the Quick Reference.
Write JAVA GUI program(s) utilizing the topics outlined here within: My 2nd GUI (or equivalent)

Complete guided notes on the String Class and its methods: (Text 2.2)
String variable creation,
Methods:

- compareTo( )
- equals( )
- indexOf( ) (both of them)
- length( )
- substring( ) (both of them)
- toLowerCase( ) & toUpperCase( )
- get & set methods for GUI Components

Enter relevant String Class methods into the Quick Reference.
Follow examples presented in class (in text & in notes)
Complete assigned homework.
Write JAVA program(s) utilizing the topics outlined here within: Address Parser (or equivalent).

Optional:
Complete guided notes on the Simple Graphics Application: (Text 1.4)
Architecture of a Graphics Application (**non**-GUI),
paintComponent( ) method,
Graphics Object and its relevant methods.
Enter relevant graphics methods into the Quick Reference.
Follow examples presented in class (in text & on supplemental handouts)
Write JAVA program(s) utilizing the topics outlined here within: Snowman (or equivalent).

**Resources:**

JAVA – An Introduction to Problem Solving & Programming: Chapter 2.2, 3 (Optional: 1.4)
Guided Notes packet (electronic)
Student Computers with NetBeans IDE, Network Drive Access & Internet Access

| Unit Modifications for Special Population Students | |
|---|---|
| **Advanced Learners** | Ask reflective and extension questions to build on classroom knowledge to develop a deeper understanding.<br>Opportunities for extended learning through additional programs (see text for possibilities).<br>Pose "What if…" questions. |
| **Struggling Learners** | Redirect student to use Quick Reference.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher.<br>Redirect student attention.<br>After school availability for help.<br>Internet resources (videos on topic, websites relevant to the particular topic, etc.). |
| **English Language Learners** | Use a translator device.<br>Provide access to language dictionary, instructor, or any other means to help interpret any language/communication difficulties.<br>Rephrase questions for student clarification.<br>Have student create vocabulary flash cards in addition to Quick Reference. |
| **Special Needs Learners** | Follow IEP modifications.<br>Redirect student attention.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher. |
| **Learners with a 504** | Refer to page four in the Parent and Educator Resource Guide to Section 504 to assist in the development of appropriate plans. |

## Interdisciplinary Connections

**Indicators:**

**ELA: RST.9-10.4; RST.11-12.4; WHST.9-10.2.D**
**Mathematics: M1, 2, 3, 4, 5, 6, 7; A-SSE.1, A-CED.2; F-LE.5**
**Science: ETS1.A: Defining and Delimiting Engineering Problems; ETS1.B: Developing Possible Solutions; ETS1.C: Optimizing the Design Solution**

## Integration of 21st Century Skills

**Indicators:**

*From the Partnership for 21st Century Skills (P21), the deeper learning competencies and skills for 21st century learning in this unit include:*

Collaboration
Communication
Critical thinking
Creativity

**Unit Title: 5) Decisions and Random Numbers**

**Unit Description:**

When analyzed, the computer essentially performs four basic tasks. By performing these tasks in combination, a computer can model ANY real-life situation. This unit focuses on one of those tasks – the Decision.

In this unit, the students will learn the various types of decision structures (If/Then, If/Then/Else, and If/Then/Else If) available in JAVA and how to apply them appropriately in a program based on the given circumstances. Compound conditions will be created through the logical operators (AND, OR, NOT). Nesting of these structures will be explored to facilitate modeling of real-life situations through more complex code.

The concept of random number generation will also be introduced. The Random Class will be used by the students to generate pseudo-random numbers whenever needed in a program. This will also necessitate the introduction of Wrapper classes and their purpose. Additionally, counters and accumulators are learned so that they can be utilized when necessary.

The conclusion of the unit focuses on the "switch" decision structure – an alternate to the If/Then/Else If structure. Students will learn the syntax and appropriate application of the switch structure as opposed to an If/Then/Else If structure. The enumeration data type will be added to the list of data types the students can access and utilize.

**Unit Duration:** 17 days

## Desired Results

**Standard(s):**

8.1 Computer Science
8.2 Design Thinking

**Indicators:**

8.1.12.CS.2: Model interactions between application software, system software, and hardware.
8.1.12.CS.3: Compare the functions of application software, system software, and hardware.
8.1.12.CS.4: Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors.
8.1.12.DA.5: Create data visualizations from large data sets to summarize, communicate, and support different interpretations of real-world phenomena.
8.1.12.DA.6: Create and refine computational models to better represent the relationships among different elements of data collected from a phenomenon or process.
8.1.12.AP.1: Design algorithms to solve computational problems using a combination of original and existing algorithms.
8.1.12.AP.3: Select and combine control structures for a specific application based upon performance and readability and identify trade-offs to justify the choice.
8.1.12.AP.5: Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
8.1.12.AP.6: Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.

8.1.12.AP.7: Collaboratively design and develop programs and artifacts for broad audiences by incorporating feedback from users.
8.1.12.AP.8: Evaluate and refine computational artifacts to make them more usable and accessible.
8.1.12.AP.9: Collaboratively document and present design decisions in the development of complex programs.

8.2.12.ED.1: Use research to design and create a product or system that addresses a problem and make modifications based on input from potential consumers.
8.2.12.ED.3: Evaluate several models of the same type of product and make recommendations for a new design based on a cost benefit analysis.
8.2.12.ED.4: Design a product or system that addresses a global problem and document decisions made based on research, constraints, trade-offs, and aesthetic and ethical considerations and share this information with an appropriate audience.
8.2.12.ED.5: Evaluate the effectiveness of a product or system based on factors that are related to its requirements, specifications, and constraints (e.g., safety, reliability, economic considerations, quality control, environmental concerns, manufacturability, maintenance and repair, ergonomics).

**Understandings:**
*Students will …*

Define Condition
List the Relational Operators and use them to create Conditions.
List the Logical Operators (AND, OR, NOT) and state their requirements.
Use the Logical Operators to create Compound Conditions.
Explain Short Circuiting of a Compound Condition.
Compare Strings with the Relational Operators & Strings with the equals( ) & compareTo( ) methods.
Write the syntax for the 3 types of If Blocks.
Write If Blocks based on a given situation (Simple Version, 2 Related Version, Several Related Version)
Nest If Blocks when dictated by a given situation.

Create a Random Number Generator (RNG) using the Random Class.
Generate random numbers using the nextInt( ) & nextDouble( ) methods.
Define Counter and Accumulator
Keep a "running total" using Counters and/or Accumulators in programs.
Define Wrapper Class & discuss their applications.
Wrap primitive data types into objects using the appropriate Wrapper class.
Convert data to Stings & integers using the toString( ) & parseInt( ) methods.
Define scope.
Create variables & objects with the correct visibility using the definition of scope & proper programming practices.

Write the syntax for the Switch statement block.

**Essential Questions:**

1. What are the situations you will encounter as a programmer that involve decisions?
2. Organize the 6 Relational Operators & the 3 Logical Operators into a table, listing their JAVA symbols and meanings.
3. What is the syntax for the decision-making structure used when there are…
    a. several unrelated choices?
    b. two related?
    c. several related?
4. What are the steps required for generating a random number in your program?
5. Differentiate between a counter and an accumulator and give a situation where each would be used.
6. What is the purpose of a Wrapper class and how is it utilized by a programmer?
7. Explain the meaning of Scope in reference to instance variables and their visibility in a Class.
8. When is a switch structure more advantageous to use over the If Then Else If structure?
9. Explain the purpose of an enumeration and outline how they are used in a program?
10. Write a program that would require the use of several separate If Then structures.
11. Provide an example of a programming situation that would require the use of an If Then Else structure.
12. Provide an example of a programming situation that would require the use of an If Then Else If structure.
13. Provide an example of a programming situation that would require the appropriate use of a Switch structure in lieu of an If Then Else If structure.

| | |
|---|---|
| Differentiate the If /Then/Else If Block with the Switch Block.<br>Write Switch Blocks when making decisions based on lists.<br>Define Enumeration.<br>Create enumerations in code & use them as appropriate (finite related lists). | |

## Assessment Evidence

| **Performance Tasks:** | **Other Evidence:** |
|---|---|
| Unit Programs (3):<br>Croupier,<br>Guess A Number,<br>Super Duper Birthday Analyzer<br>(Or equivalents) | **Formative Assessments:**<br>Unit Quizzes (2)<br><br>**Summative Assessment(s)**<br>Unit Test |

**Benchmarks:**

Super Duper Birthday Analyzer program

## Learning Plan

**Learning Activities:**

Complete guided notes on If Then Structures: (Text 3.1, 2)
- Conditions & Compound Conditions,
- Relational & Logical Operators,
- Short Circuiting
- Comparing Strings,
- If Then structures (If Then, If Then Else, If Then Else If),
- Appropriate application of the 3 different If Then Structures,
- Nesting If Then Structures.

Enter the If Then structure (all 3 versions) into the Quick Reference.
Follow examples presented in class (in text & on supplemental handouts).
Complete assigned homework.
Write JAVA program(s) utilizing the topics outlined here within: Croupier (or equivalent).

Complete guided notes Random Numbers:
- Random Class, creation of a random number generator, use of its methods.
- Counters & Accumulators, appropriate application of Counters & Accumulators in a programming situation.
- Wrapper classes, conversion methods for converting from numbers to String values and vice versa.
- Scope (class level vs. local variables and their appropriate applications).

Enter the Random Class nextInt( ) & nextDouble( ) methods, the Wrapper Class methods, as well as the parseInt( ) method into the Quick Reference.
Follow examples presented in class (in text & in notes).
Complete assigned homework.
Write JAVA program(s) utilizing the topics outlined here within: Guess A Number (or equivalent)

Complete guided notes on the Switch decision structure and Enumerations: (Text 3.3)
- Switch decision structure and its application (alternate to If Then Else If when using lists).
- Enumerations (creation and use).

Enter the switch decision structure as well as the enumeration syntax into the Topical Index.
Follow examples presented in class (in text & on supplemental handouts).

Complete assigned homework.
Write JAVA program(s) utilizing the topics outlined here within: Super Duper Birthday Analyzer (or equivalent).

**Resources:**

JAVA – An Introduction to Problem Solving & Programming: Chapter 3.1-3
Guided Notes packet (electronic)
Student Computers with NetBeans IDE, Network Drive Access & Internet Access

| Unit Modifications for Special Population Students | |
|---|---|
| **Advanced Learners** | Ask reflective and extension questions to build on classroom knowledge to develop a deeper understanding.<br>Opportunities for extended learning through additional programs (see text for possibilities).<br>Pose "What if…" questions. |
| **Struggling Learners** | Redirect student to use Quick Reference.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher.<br>Redirect student attention.<br>After school availability for help.<br>Internet resources (videos on topic, websites relevant to the particular topic, etc.). |
| **English Language Learners** | Use a translator device.<br>Provide access to language dictionary, instructor, or any other means to help interpret any language/communication difficulties.<br>Rephrase questions for student clarification.<br>Have student create vocabulary flash cards in addition to Quick Reference. |
| **Special Needs Learners** | Follow IEP modifications.<br>Redirect student attention.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher. |
| **Learners with a 504** | Refer to page four in the Parent and Educator Resource Guide to Section 504 to assist in the development of appropriate plans. |

## Interdisciplinary Connections

**Indicators:**
**ELA: RST.9-10.4,7; RST.11-12.4,7; WHST.9-10.2.D**
**Mathematics: M1, 2, 3, 4, 5, 7, 8**
**Science: ETS1.A: Defining and Delimiting Engineering Problems; ETS1.B: Developing Possible Solutions; ETS1.C: Optimizing the Design Solution**

## Integration of 21st Century Skills

**Indicators:**

*From the Partnership for 21st Century Skills (P21), the deeper learning competencies and skills for 21st century learning in this unit include:*

Collaboration
Communication

Critical thinking
Creativity

Unit Title: 6) Loops

**Unit Description:**

This unit focuses on another of the aforementioned “four basic tasks a computer can perform” – the Loop (or Iteration).

In this unit the students will learn the various types of loops (While, Do While, & For Loop) available in JAVA, how to nest them, and how to apply them appropriately in a program.

**Unit Duration:** 19 days

## Desired Results

**Standard(s):**

8.1 Computer Science
8.2 Design Thinking

**Indicators:**

8.1.12.CS.2: Model interactions between application software, system software, and hardware.
8.1.12.CS.3: Compare the functions of application software, system software, and hardware.
8.1.12.CS.4: Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors.
8.1.12.AP.1: Design algorithms to solve computational problems using a combination of original and existing algorithms.
8.1.12.AP.3: Select and combine control structures for a specific application based upon performance and readability and identify trade-offs to justify the choice.
8.1.12.AP.4: Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue.
8.1.12.AP.5: Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
8.1.12.AP.6: Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
• 8.1.12.AP.7: Collaboratively design and develop programs and artifacts for broad audiences by incorporating feedback from users.
8.1.12.AP.8: Evaluate and refine computational artifacts to make them more usable and accessible.
8.1.12.AP.9: Collaboratively document and present design decisions in the development of complex programs.

8.2.12.ED.1: Use research to design and create a product or system that addresses a problem and make modifications based on input from potential consumers.

8.2.12.ED.5: Evaluate the effectiveness of a product or system based on factors that are related to its requirements, specifications, and constraints (e.g., safety, reliability, economic considerations, quality control, environmental concerns, manufacturability, maintenance and repair, ergonomics).
8.2.12.NT.2: Redesign an existing product to improve form or function.

**Understandings:**
*Students will …*

Write the syntax for the While loop.
Write the syntax for the Do While loop.
Recite when to use the 2 types of While loops (…when number of repetitions IS NOT known)
Differentiate the While and Do While loops and use them appropriately (whether loop body **may or may not be executed** vs. **at least once**).
Define sentinel & use sentinels to direct the flow of loops appropriately.
Define Infinite loop and know how to avoid/correct them within programs.
Define nested loop & write nested loops as appropriate in programs.

Write a For loop using proper syntax.
State when to use a For Loop (number of iterations IS known).
Direct the flow of a For Loop by appropriately initializing and setting the condition & increment.
Nest For Loops according to program requirements.

**Essential Questions:**

1. What part of your life is repeated?
   a) Are there situations when a computer will have to perform a task over & over again? What are they?
2. What are the two subtypes of loops used when the number of iterations is NOT known? Provide a situation in which each would be correctly applied.
3. Explain what a sentinel is and how is it used to direct the flow of a loop?
4. What is the type of loop used when the number of iterations IS known? Provide a situation in which this type of loop would be correctly applied.
5. Describe a situation where you would need to perform multiple loops (loops within loops).
6. What is an infinite loop, what are its causes, and what are its remedies?
7. Write a program that would require a user to input information (grades, courses, etc.) over & over again until they indicate they have finished entering information.
8. Write a program that would require searching for a user-selected character in a string repeatedly.
9. Write a program that would display the Unicode Character set from 32 to 126.
10. Write a program that would display a Multiplication Table for the numbers 0 through 9.

## Assessment Evidence

**Performance Tasks:**

Unit Programs (4):
Guess-A-Number Rewrite,
Poor Man's Wheel of Fortune,
Unicode Chart and Multiplication Chart
(Or equivalents)

**Other Evidence:**

**Formative Assessments:**
Unit Quizzes (2)

**Summative Assessment(s)**
Unit Test

**Benchmarks:**

Multiplication Chart

## Learning Plan

**Learning Activities:**

Complete guided notes on While, Do While, & Nested Loops: (Text 4.1-2)

The 2 types of Loop situations (number of repetitions **IS** known or **IS NOT** known)

First, While loops (number of repetitions **IS NOT** known)

- While Loop structure/syntax
- Do While Loop structure/syntax
- flow
- appropriate application of each
- Sentinels
- infinite loops
- nesting

Enter the While Loop types into the Quick Reference.
Follow examples presented in class (in text & in notes).
Copy & run a small sample program utilizing While loops.
Complete assigned homework.
Write JAVA program(s) utilizing the types of While loops learned: Guess-A-Number Rewrite, Poor Man's Wheel of Fortune (or equivalents)

Complete guided notes on For Loops: (Text 4.1-2)

- number of repetitions **IS** known
- control variable, increment, flow
- Nesting

Enter the For Loop type into the Quick Reference.
Follow examples presented in class (in text & in notes).
Complete assigned homework.
Write JAVA program(s) utilizing the types of For loops learned: Unicode Chart, Multiplication Table (or equivalents)

**Resources:**

JAVA – An Introduction to Problem Solving & Programming: Chapter 4.1-2
Guided Notes packet (electronic)
Student Computers with NetBeans IDE, Network Drive Access & Internet Access

| Unit Modifications for Special Population Students | |
|---|---|
| **Advanced Learners** | Ask reflective and extension questions to build on classroom knowledge to develop a deeper understanding.<br>Opportunities for extended learning through additional programs (see text for possibilities).<br>Pose "What if…" questions. |
| **Struggling Learners** | Redirect student to use Quick Reference.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher.<br>Redirect student attention.<br>After school availability for help.<br>Internet resources (videos on topic, websites relevant to the particular topic, etc.). |
| **English Language Learners** | Use a translator device.<br>Provide access to language dictionary, instructor, or any other means to help interpret any language/communication difficulties.<br>Rephrase questions for student clarification.<br>Have student create vocabulary flash cards in addition to Quick Reference. |
| **Special Needs Learners** | Follow IEP modifications.<br>Redirect student attention.<br>Rephrase questions for student clarification. |

| | |
|---|---|
| | Preferential seating – close proximity to teacher. |
| **Learners with a 504** | Refer to page four in the Parent and Educator Resource Guide to Section 504 to assist in the development of appropriate plans. |

## Interdisciplinary Connections

**Indicators:**

**ELA: RST.9-10.4; RST.11-12.4; WHST.9-10.2.D**
**Mathematics: M1, 2, 3, 4, 5, 6, 7, 8**
**Science: ETS1.A: Defining and Delimiting Engineering Problems; ETS1.B: Developing Possible Solutions; ETS1.C: Optimizing the Design Solution**

## Integration of 21st Century Skills

**Indicators:**

*From the Partnership for 21st Century Skills (P21), the deeper learning competencies and skills for 21st century learning in this unit include:*

Collaboration
Communication
Critical thinking
Creativity

**Unit Title: 7) Writing Custom Classes**

**Unit Description:**

This unit moves the students into OOP programming. The focus switches from "procedural" programming toward modeling real world situations by creating *objects* which model those situations. This is a fundamental paradigm shift.

The students have, up until now, always utilized Classes provided by the JAVA Standard Class Library. In no way can these existing Classes model **every** situation that will be encountered by a programmer. Therefore, the student programmer must be able to branch out and create their own **custom** Classes in order to reach full potential. This is a cornerstone of OOP programming.

Students will re-examine the "anatomy" of a class in much greater detail. This will require expanding previous concepts: instance variables, constructors, methods. New concepts will also need to be introduced: accessors, mutators, passing returning values and the toString( ) method. The need for the creation of a "driver" program to test out their custom class will also be explored.

Following this program, the students will learn the importance of documenting with Pre & Post Conditions, how to overload methods properly, and Decomposition.

**Unit Duration: 28 days**

## Desired Results

**Standard(s):**

8.1 Computer Science
8.2 Design Thinking

**Indicators:**

8.1.12.CS.2: Model interactions between application software, system software, and hardware.
8.1.12.CS.3: Compare the functions of application software, system software, and hardware.
8.1.12.CS.4: Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors.
8.1.12.AP.1: Design algorithms to solve computational problems using a combination of original and existing algorithms.
8.1.12.AP.2: Create generalized computational solutions using collections instead of repeatedly using simple variables.
8.1.12.AP.3: Select and combine control structures for a specific application based upon performance and readability and identify trade-offs to justify the choice.
8.1.12.AP.4: Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue.
8.1.12.AP.5: Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
8.1.12.AP.6: Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
8.1.12.AP.7: Collaboratively design and develop programs and artifacts for broad audiences by incorporating feedback from users.
8.1.12.AP.8: Evaluate and refine computational artifacts to make them more usable and accessible.

8.1.12.AP.9: Collaboratively document and present design decisions in the development of complex programs.

8.2.12.ED.5: Evaluate the effectiveness of a product or system based on factors that are related to its requirements, specifications, and constraints (e.g., safety, reliability, economic considerations, quality control, environmental concerns, manufacturability, maintenance and repair, ergonomics).
8.2.12.NT.1: Explain how different groups can contribute to the overall design of a product.

**Understandings:**
*Students will …*

Describe the different sections of an object class
- Naming conventions
- Comment Header
- Data (Instance variables)
- Constructor
- Methods, Parameters & Arguments

Use a custom written class by importing it, creating an object from it & calling the included methods.
Write a driver program to test custom written classes.
Create accessor & mutator methods to get/set instance variables.
Create a toString( ) method for custom classes.

Define, identify & write Pre & Post-Conditions for methods in a custom class.
Overload methods by creating constructors/methods with varying signatures.
Define Decomposition.
Create complex classes by applying the concept of Decomposition.

**Essential Questions:**

1. Outline the compete anatomy of a JAVA custom class.
    a. syntactical framework
    b. comment headers (both overall class and individual methods)
    c. instance variables (& how are instance variables encapsulated)
    d. constructor
    e. methods (including accessors, mutators and the toString( ) method)
2. Outline the steps required to create a custom class and integrate it into a program.
3. Why do we create custom Classes rather than simply entering code into a single main( ) method?
4. Explain the meaning of a driver and how is it utilized with the creation of custom JAVA classes?
5. Justify the need for Pre- and Post-Conditions.
6. Why would a programmer need to overload a method?
7. Describe the concept of Decomposition and relate how it can aid in writing complex programs.
8. Write a custom class that would model a real-world object and write a driver program to fully test it.
9. Write several custom classes that would model the required elements of a solitaire game and write a driver program to fully test it.

## Assessment Evidence

**Performance Tasks:**

Unit Programs (4):

Coin & Die Class copied from text,

Create a driver to test those classes,

Create a Spinner Class and update driver,

Create a Twister Spinner Class and update driver,

(Or equivalents)

Model the “Game of Pig” solitaire game with a custom class (from scratch) & update driver

(Or equivalents)

**Other Evidence:**

**Formative Assessments:**
Unit Quizzes (2)

**Summative Assessment(s)**

Unit Test

| | |
|---|---|
| | |

**Benchmarks:**

Game of Pig

## Learning Plan

**Learning Activities:**

Complete guided notes on Writing Custom Classes: (Text 5.1-2)
- Brief review of the Anatomy of a JAVA Class.
- In depth explanations of Anatomy of a JAVA Class:
  - comment header(s)
  - structure / syntax
  - naming conventions
  - instance variables (declaration, no initialization / instantiation, scope)
  - constructor (initialization / instantiation of instance variables done within, with and without parameters)
  - methods:
    - with and without parameters
    - returning a value from
    - accessors, mutators, toString( ) methods
- Full process of creating a custom class and incorporating it into a program (imports, object creation, methods use).
- Creation of a driver program for testing.

Follow examples presented in class (in text & in notes).
Update the Quick Reference as necessary.
Complete assigned homework.
Copy the Coin & Die Classes provide in the notes.
Write JAVA program(s) utilizing topics outlined here within:
- Create a driver program to test the Coin & Die Classes
- Create a Spinner Class and update the previous driver program to include it.
- Create a Twister Class and update the previous driver program to include it (or equivalents).

Complete guided notes on Enhancing Custom Classes: (Text 5.1-2)
- Pre & Post Conditions,
- Overloading for use with multiple data types in parameters,
- Decomposition of complex problems into simpler manageable elements.

Follow examples presented in class (in notes).
Update the Quick Reference as necessary.
Complete assigned homework.
Write JAVA program(s) utilizing the topics outlined here within:
- Create a program simulating the solitaire dice game of Pig (or equivalent).

**Resources:**

JAVA – An Introduction to Problem Solving & Programming: Chapter 5.1-5.2
Guided Notes packet (electronic)
Student Computers with NetBeans IDE, Network Drive Access & Internet Access

| Unit Modifications for Special Population Students | |
|---|---|
| **Advanced Learners** | Ask reflective and extension questions to build on classroom knowledge to develop a deeper understanding.<br>Opportunities for extended learning through additional programs (see text for possibilities).<br>Pose "What if…" questions. |
| **Struggling Learners** | Redirect student to use Quick Reference.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher.<br>Redirect student attention.<br>After school availability for help.<br>Internet resources (videos on topic, websites relevant to the particular topic, etc.). |
| **English Language Learners** | Use a translator device.<br>Provide access to language dictionary, instructor, or any other means to help interpret any language/communication difficulties.<br>Rephrase questions for student clarification.<br>Have student create vocabulary flash cards in addition to Quick Reference. |
| **Special Needs Learners** | Follow IEP modifications.<br>Redirect student attention.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher. |
| **Learners with a 504** | Refer to page four in the Parent and Educator Resource Guide to Section 504 to assist in the development of appropriate plans. |

## Interdisciplinary Connections

**Indicators:**

**ELA: RST.9-10.4; RST.11-12.4; WHST.9-10.2.D**
**Mathematics: M1, 2, 3, 4, 5, 7,8**
**Science: ETS1.A: Defining and Delimiting Engineering Problems; ETS1.B: Developing Possible Solutions; ETS1.C: Optimizing the Design Solution**

## Integration of 21st Century Skills

**Indicators:**

*From the Partnership for 21st Century Skills (P21), the deeper learning competencies and skills for 21st century learning in this unit include:*

Collaboration
Communication
Critical thinking
Creativity

**Unit Title: 8) Passing Variables & Objects, Static, 1D Arrays**

**Unit Description:**

This unit begins with enhancing the student's programming ability by deepening the understanding of how information is passed between various objects within a program. The related concept of an alias is also introduced. The concept of, and need for, static variables & methods is also studied in depth.

The unit concludes with the concept of an array (one-dimensional only). The students will learn how to create single dimension arrays of various variable types and how to initialize them, access them, alter them, and to determine their size. An additional type of loop will also be studied, the For…Each loop. How the For…Each loop is specifically suited for arrays will be learned.

**Unit Duration: 15 days**

## Desired Results

**Standard(s):**

8.1 Computer Science
8.2 Design Thinking

**Indicators:**

8.1.12.CS.1: Describe ways in which integrated systems hide underlying implementation details to simplify user experiences.
8.1.12.CS.2: Model interactions between application software, system software, and hardware.
8.1.12.CS.3: Compare the functions of application software, system software, and hardware.
8.1.12.CS.4: Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors.
8.1.12.DA.2: Describe the trade-offs in how and where data is organized and stored.
8.1.12.DA.5: Create data visualizations from large data sets to summarize, communicate, and support different interpretations of real-world phenomena.
8.1.12.DA.6: Create and refine computational models to better represent the relationships among different elements of data collected from a phenomenon or process.
8.1.12.AP.1: Design algorithms to solve computational problems using a combination of original and existing algorithms.
8.1.12.AP.2: Create generalized computational solutions using collections instead of repeatedly using simple variables.
8.1.12.AP.3: Select and combine control structures for a specific application based upon performance and readability and identify trade-offs to justify the choice.
8.1.12.AP.4: Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue.
8.1.12.AP.5: Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
8.1.12.AP.6: Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
8.1.12.AP.8: Evaluate and refine computational artifacts to make them more usable and accessible.
8.1.12.AP.9: Collaboratively document and present design decisions in the development of complex programs.

8.2.12.ED.1: Use research to design and create a product or system that addresses a problem and make modifications based on input from potential consumers.
8.2.12.ED.5: Evaluate the effectiveness of a product or system based on factors that are related to its requirements, specifications, and constraints (e.g., safety, reliability, economic considerations, quality control, environmental concerns, manufacturability, maintenance and repair, ergonomics).
8.2.12.NT.1: Explain how different groups can contribute to the overall design of a product.

**Understandings:**
*Students will …*

Define **alias** & describe how they are used in a program.
Define "Passing by Value" and its ramifications on **primitive** types & **objects** in a program.
Describe how the **static** modifier alters how variables and methods are accessed in objects/classes

Define & describe an Array (1D).
Declare arrays using a constructor and through an Initializer List.
Assign and/or access elements using indices.
Use "traditional" loops (previous material) to initialize and/or access array elements.
Use the specialized "For…Each" loop to access the contents of an entire array.
Pass arrays as parameters.

**Essential Questions:**

1. Provide an example of how an alias could be used in a programming situation? How are the variable contents affected when the original variable and alias are changed?
2. Diagram exactly what occurs to primitive & object variable types when information is passed.
3. What effect does using the "static" visibility modifier have on a variable? A method?
4. Why use static variables and methods? In what situations would they apply?
5. What variable name(s) would you use to hold the names of all the students in this high school?
    a. Is that practical?
    b. Relate this to what the code would look like if you were to try and calculate an average of all those grades?
6. Explain the process of creating and using a 1D array
    a. Creation (both ways)
    b. Accessing individual elements
    c. Altering individual elements
    d. Displaying the contents of an entire array all at once
7. What is a For…Each loop?
    a. Justify its need when dealing with arrays?
    b. Outline the For…Each loop's limitations?
8. Write a program that employ the use of aliases, static variables & methods.
9. Provide 3 examples of a programming situation that would require the use of a 1D array.

## Assessment Evidence

**Performance Tasks:**

Unit Programs (2):
Static Demo,
Quarterback Ratings Calculator

**Other Evidence:**

**Formative Assessments:**
Unit Quizzes (2)

**Summative Assessment(s)**
Unit Test

**Benchmarks:**

Quarterback Ratings Calculator

## Learning Plan

**Learning Activities:**

Complete guided notes on: (Text 5.3, 6.2)

Aliases,

Passing

- by Value
- effect on variable content with a primitive variable type
- effect on variable content with an object variable type

The Static modifier

- meaning
- effect on variables and their proper use
- effect on methods and their proper use.

Enter the syntax for the *static* modifier into the Quick Reference.
Follow examples presented in class (in text & in notes).
Complete assigned homework.
Write JAVA program(s) utilizing the types of loops learned: Static Demo (or equivalent)

Complete guided notes on Arrays (one dimensional) and the For…Each loop: (Text 7.1-2)

1D Arrays

- creation (constructor and initializer lists)
- access
  - individual elements
  - using a standard For loop to access the entire loop at once (including removal and replacement)
  - using a For…Each loop to access the entire loop at once (**retrieval only**)
- passing arrays

Enter the syntax for array declaration (both ways), For…Each Loop syntax, and the syntax for passing an array as a parameter into the Quick Reference.
Follow examples presented in class (in text & in notes).
Complete assigned homework.
Write JAVA program(s) utilizing the types of loops learned: Quarterback Ratings Calculator (or equivalent).

**Resources:**

JAVA – An Introduction to Problem Solving & Programming: Chapter 5.3, 6.2 & 7.1-2
Guided Notes packet (electronic)
Student Computers with NetBeans IDE, Network Drive Access & Internet Access

| Unit Modifications for Special Population Students | |
|---|---|
| **Advanced Learners** | Ask reflective and extension questions to build on classroom knowledge to develop a deeper understanding.<br>Opportunities for extended learning through additional programs (see text for possibilities).<br>Pose "What if…" questions. |
| **Struggling Learners** | Redirect student to use Quick Reference.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher.<br>Redirect student attention.<br>After school availability for help.<br>Internet resources (videos on topic, websites relevant to the particular topic, etc.). |
| **English Language Learners** | Use a translator device.<br>Provide access to language dictionary, instructor, or any other means to help interpret any language/communication difficulties.<br>Rephrase questions for student clarification.<br>Have student create vocabulary flash cards in addition to Quick Reference. |
| **Special Needs Learners** | Follow IEP modifications.<br>Redirect student attention.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher. |
| **Learners with a 504** | Refer to page four in the Parent and Educator Resource Guide to Section 504 to assist in the development of appropriate plans. |

## Interdisciplinary Connections

**Indicators:**

**ELA: RST.9-10.4; RST.11-12.4; WHST.9-10.2.D**
**Mathematics: M1, 2, 3, 4, 5, 6, 7, 8; A-CED, F-LE.5, S-ID, S-MD**
**Science: ETS1.A: Defining and Delimiting Engineering Problems; ETS1.B: Developing Possible Solutions; ETS1.C: Optimizing the Design Solution**

## Integration of 21st Century Skills

**Indicators:**

*From the Partnership for 21st Century Skills (P21), the deeper learning competencies and skills for 21st century learning in this unit include:*

Collaboration
Communication
Critical thinking
Creativity

**Unit Title: 9) Sorting Algorithms**

**Unit Description:**

The final unit introduces the student to the essential concept of Sorting. Two algorithms for sorting will be introduced, the Selection Sort & Insertion Sort. Students will be able to apply these algorithms to sort arrays.

**Unit Duration: 6 days**

## Desired Results

**Standard(s):**

8.1 Computer Science
8.2 Design Thinking

**Indicators:**

8.1.12.CS.1: Describe ways in which integrated systems hide underlying implementation details to simplify user experiences.
8.1.12.CS.2: Model interactions between application software, system software, and hardware.
8.1.12.CS.3: Compare the functions of application software, system software, and hardware.
8.1.12.CS.4: Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors.
8.1.12.DA.1: Create interactive data visualizations using software tools to help others better understand real world phenomena, including climate change.
8.1.12.DA.2: Describe the trade-offs in how and where data is organized and stored.
8.1.12.DA.6: Create and refine computational models to better represent the relationships among different elements of data collected from a phenomenon or process.
8.1.12.AP.1: Design algorithms to solve computational problems using a combination of original and existing algorithms.
8.1.12.AP.2: Create generalized computational solutions using collections instead of repeatedly using simple variables.
• 8.1.12.AP.3: Select and combine control structures for a specific application based upon performance and readability and identify trade-offs to justify the choice.
8.1.12.AP.4: Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue.
8.1.12.AP.5: Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
8.1.12.AP.6: Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
8.1.12.AP.7: Collaboratively design and develop programs and artifacts for broad audiences by incorporating feedback from users.
8.1.12.AP.8: Evaluate and refine computational artifacts to make them more usable and accessible.
8.1.12.AP.9: Collaboratively document and present design decisions in the development of complex programs.

8.2.12.ED.1: Use research to design and create a product or system that addresses a problem and make modifications based on input from potential consumers.

8.2.12.ED.3: Evaluate several models of the same type of product and make recommendations for a new design based on a cost benefit analysis.
8.2.12.ED.5: Evaluate the effectiveness of a product or system based on factors that are related to its requirements, specifications, and constraints (e.g., safety, reliability, economic considerations, quality control, environmental concerns, manufacturability, maintenance and repair, ergonomics).
8.2.12.NT.2: Redesign an existing product to improve form or function.

| **Understandings:**<br>*Students will …*<br><br>Explain the Insertion & Selection Sort Algorithms.<br>Apply the sorting algorithms in a JAVA program. | **Essential Questions:**<br><br>1. Outline the algorithm behind the<br>a. Selection Sort<br>b. Insertion Sort<br>2. Write a program that would sort the Quarterback ratings from the previous program using the Insertion & Selection Sort Algorithms. |
|---|---|

## Assessment Evidence

| **Performance Tasks:**<br><br>Unit Program:<br>Quarterback Ratings Calculator Update (to sorting) | **Other Evidence:** |
|---|---|

**Benchmarks:**

Quarterback Ratings Calculator Update

## Learning Plan

**Learning Activities:**

Complete guided notes on Sorting Algorithms: (Text 7.4)
- Selection Sort
- Insertion Sort
- Animations of Selection & Insertion sort Algorithms (Internet websites).

Enter the Insertion & Selection Sort code (copied from text) into the Quick Reference.
Follow examples presented in class (in text & in notes).
Complete assigned homework.
Write JAVA program(s) utilizing the types of loops learned: Quarterback Ratings Calculator Update (to sorting)
(Or equivalent)

**Resources:**

JAVA – An Introduction to Problem Solving & Programming: Chapter 7.4
Guided Notes packet (electronic)
Student Computers with NetBeans IDE, Network Drive Access & Internet Access

## Unit Modifications for Special Population Students

| | |
|---|---|
| **Advanced Learners** | Ask reflective and extension questions to build on classroom knowledge to develop a deeper understanding.<br>Opportunities for extended learning through additional programs (see text for possibilities).<br>Pose “What if…” questions. |
| **Struggling Learners** | Redirect student to use Quick Reference.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher. |

| | |
|---|---|
| | Redirect student attention.<br>After school availability for help.<br>Internet resources (videos on topic, websites relevant to the particular topic, etc.). |
| **English Language Learners** | Use a translator device.<br>Provide access to language dictionary, instructor, or any other means to help interpret any language/communication difficulties.<br>Rephrase questions for student clarification.<br>Have student create vocabulary flash cards in addition to Quick Reference. |
| **Special Needs Learners** | Follow IEP modifications.<br>Redirect student attention.<br>Rephrase questions for student clarification.<br>Preferential seating – close proximity to teacher. |
| **Learners with a 504** | Refer to page four in the Parent and Educator Resource Guide to Section 504 to assist in the development of appropriate plans. |

## Interdisciplinary Connections

**Indicators:**

**ELA: RST.9-10.4; RST.11-12.4; WHST.9-10.2.D**
**Mathematics: M1, 2, 3, 4, 5, 6, 7, 8; A-CED, F-LE.5, S-ID, S-MD**
**Science: ETS1.A: Defining and Delimiting Engineering Problems; ETS1.B: Developing Possible Solutions; ETS1.C: Optimizing the Design Solution**

## Integration of 21st Century Skills

**Indicators:**

*From the Partnership for 21st Century Skills (P21), the deeper learning competencies and skills for 21st century learning in this unit include:*

Collaboration
Communication
Critical thinking
Creativity