Curriculum Guide
Program Area: Software Development

Standard 1.0 - APPLY PROBLEM-SOLVING AND CRITICAL THINKING SKILLS TO INFORMATION TECHNOLOGY					
Measurement Criteria		**Concepts**	**Implementation**	**Terminology**	**Testing Item**
1.1	Describe methods and considerations for prioritizing and scheduling software development tasks (e.g., riskvalue, waterfall, agile, GTD, Kanban)		Describe why one should speak to the customer before starting computer repairs	Sequential Design Process	X
			Describe benefits of following a sequential design process.	Waterfall GTD	
1.2	Describe methods and techniques of problem-solving and troubleshooting applicable to software development		Describe the steps in solving a computer problem (customer service) Describe the purpose of a work order and what information is included on a work order	Work order	X
				Program Change Request	

Standard 2.0 - MAINTAIN A SAFE GREEN INFORMATION TECHNOLOGY ENVIRONMENT					
Measurement Criteria		**Concepts**	**Implementation**	**Terminology**	**Testing Item**
2.1	Identify ergonomics and repetitive strain injuries common in information technology		Explain why the individual is ultimately responsible for maintaining a safe and healthy workplace Identify where to dispose of CRT monitors Discuss ways to make a computer environment environmentally friendly	Carpel tunnel syndrome	X

			Explain the environmental considerations when disposing of networking components		
2.2	Describe programming techniques to analyze and reduce power consumption (e.g., Algorithmic Efficiency, Big O notation)				X
			Explain how Algorithmic Efficiency and Big O notation is used to make code efficient Explain the value of an efficient algorithm	Algorithmic Efficiency Big O notation	
2.3	Describe the environmental importance of the proper disposal of computer components, including safe disposal of toxic chemicals				X
			Explain why proper care should be taken when disposing of a computer or device (phones, tablets etc.)	E-Trash/Waste	
2.4	Explain the impact of large scale computing (e.g., power consumption, thermal impact)				X
			Discuss with students what environmental impacts can large scale computing have on an area or a building (electricity, HVAC, and general design)		

Standard 3.0 - RECOGNIZE GENERAL SECURITY ISSUES RELATED TO INFORMATION TECHNOLOGY				
Measurement Criteria	**Concepts**	**Implementation**	**Terminology**	**Testing Item**

This project partially funded in Partnership with AZ CTE Programs of Study

3.1	Explain procedures to maintain data integrity and security (e.g., lock the screen, don't open unrecognized emails, don't plug in untrusted thumb drives, only use approved software)				X
			Explain practices computer users can implement to protect their personal data, business data, Explain purpose of locking your computer screen Explain the indicators that would alert a user to a possible email threat.		
3.2	Identify security issues related to the network, computer hardware, software, and data		Explain the purpose of a fire wall Describe how software updates and anti-virus software protect the computer	Anti-virus software Firewall	X
			Describe the importance of applying regular software updates		
3.3	Describe computer threats and methods to protect a computer (e.g., viruses, phishing, e-mail, social engineering, spoofing, identify theft, spamming				X
			Explain what ways technology has enabled social engineering Explain the ways to protect yourself against a computer virus	Viruses Phishing Social Engineering Spoofing Identify Theft Spamming	

This project partially funded in Partnership with AZ CTE Programs of Study

3.4	Explain concepts such as denial of service, hacking/cracking, intrusion, detection, and prevention		Explain distributed denial of service		X
			Describe benefits to learning hacking methods Describe the dangers and benefits to hacking	Hacking White Collar Hacking Software Cracking Intrusion Detection	

Standard 4.0 - RECOGNIZE SPECIFIC SECURITY ISSUES RELATED TO SOFTWARE DEVELOPMENT					
Measurement Criteria		**Concepts**	**Implementation**	**Terminology**	**Testing Item**
4.1	Explain best practices for maintaining integrity and security in software development (e.g., encryption, hashing, digital signatures)				X
			List (in order of simple to sophisticated) various methods of data security. Explain why hashing improve data security. Describe how have digital signatures improved business security	Data Integrity Encryption Hashing Digital Signatures	
4.2	Describe methods for sanitizing user input to prevent issues such as buffer overflows and SQL injection			User input validation sanitizing	X
			Explain the importance of validating user input data to prevent unauthorized access	Buffer Overflow SQL Injection	

This project partially funded in Partnership with AZ CTE Programs of Study

			Provide examples of problems that could occur without proper data entry validations		
4.3	Explain the difference between authentication and authorization				X
			Give an example of a computer user who is legitimately authenticated but not authorized.	Authentication Authorization	
4.4	Explain how software defects relate to software security (e.g., buffer overflows, cross-site scripting)			XSS (Cross-site scripting).	x
			XSS (Cross-site scripting).		

Standard 5.0 - EXPLORE LEGAL AND ETHICAL ISSUES RELATED TO INFORMATION TECHNOLOGY					
Measurement Criteria		**Concepts**	**Implementation**	**Terminology**	**Testing Item**
5.1	Explore intellectual property rights including software licensing and software duplication		Describe software licensing restrictions Describe the legal use for a copy of non- open source personal software	Unethical	X
				Software license	
5.2	Explain legal and ethical issues related to the difference between various open source and proprietary licenses (e.g., GPL, BSD, and proprietary licenses)			Open Source	X
			Explain the benefits and disadvantages of licensing	General Public License	

This project partially funded in Partnership with AZ CTE Programs of Study

			Explain the benefits and disadvantages of open source		
5.3	Identify issues and trends affecting computers and information privacy (e.g., HIPAA, COPPA, CISPA, FERPA, data brokers)				X
			Explain your responsibility with the respect to HIPAA, COPPA, CISPA, FERPA, data brokers.	HIPAA COPPA CISPA FERPA Data Brokers	
5.4	Describe licensing considerations for using third party libraries when creating software for sale (e.g. opensource restrictions, proprietary restrictions, copyleft licenses)				X
			Explain the disadvantages of third party libraries	Copyleft Third Party Libraries	
5.5	Differentiate between ethical and legal uses of information technology (e.g., data pricing, use of public and private networks, social networking, industry-related data, data piracy)				X
			Differentiate between legal and ethical use of information (provide examples)	Data Pricing, Social Networking Industry-Related Data Data Piracy	

Standard 6.0 - DEMONSTRATE BASIC COMPUTER MATHEMATICS REQUIRED FOR INFORMATION TECHNOLOGY

This project partially funded in Partnership with AZ CTE Programs of Study

Measurement Criteria		Concepts	Implementation	Terminology	Testing Item
6.1	Explain the function of base number systems in mathematics as they relate to computer technology		Describe the logical operation, AND, OR and NOT		X
			Describe why computers function with a base of 2	Binary Hexadecimal	
6.2	Perform decimal to binary and binary to decimal conversions		Identify the largest number an eight bit byte can hold Determine the decimal (Base Ten) equivalent of the binary number when given a problem Determine the binary equivalent when given a decimal (Base Ten)		X
			Solve conversions		
6.3	Perform decimal to hexadecimal and hexadecimal to decimal conversions				X
			Solve conversions		
6.4	Perform hexadecimal to binary and binary to hexadecimal conversions		Calculate the Binary equivalent of a Hexadecimal number		X
			Solve conversions		
6.5	Determine the appropriate method to perform conversions (e.g., paper/pencil, electronic resources)				X
			When converting decimal to binary the best method is paper pencil		
6.6	Perform modulo and bitwise operations		Given a problem identify the type of data function that is produced	Modulo Bitwise	X

This project partially funded in Partnership with AZ CTE Programs of Study

			Calculate Modulus given a problem Calculate Bitwise given a problem		

Standard 7.0 - DESCRIBE THE DEVELOPMENT/EVOLUTION OF COMPUTERS AND INFORMATION TECHNOLOGY

Measurement Criteria		Concepts	Implementation	Terminology	Testing Item
7.1	1 Describe a computer and its components and functions		Identify the procedure a computer uses to process data Describe the purpose of surge protectors Identify technology used for output (smart phone, computer monitor, printer) Explain the purpose of a CPU Explain the purpose of a keyboard Identify whether a computer component functions as an input or an output		X
7.2	Describe the evolution of the computer through its generations (e.g., vacuum tubes, transistors, integrated circuits)		Identify how many bits make a byte Identify what replaced vacuumed tubes in the next generation computer Identify the component that led to the creation of a home computer	Binary Transistors Miniaturization Microprocessor Vacuum Tubes Integrated Circuits	X
			Explain why technology has grown so quickly (smaller and faster)	Voice Recognition	

This project partially funded in Partnership with AZ CTE Programs of Study

7.3	Explain the evolution of programming languages from low level to high level (e.g., machine code, assembly, higher-level languages)				X
			Describe how programming used to be a manual input process Describe OOP (Object oriented programing) Identify examples of machine code, assemble code and higher level languages	machine code, assembly, higher-level languages	
7.4	Explain how the development of computers has impacted modern life		Explain the societal impact of computers and technology. (health, third world countries, space exploration)		X
7.5	Identify future trends in computing (e.g., quantum, optical, DNA)				X
			Describe how and why these trends will impact the society Describe how wireless data transmission has changed and will change society	Quantum Optical DNA Satellite	

Standard 8.0 - USE VERSION CONTROL SYSTEMS

Measurement Criteria	Concepts	Implementation	Terminology	Testing Item

This project partially funded in Partnership with AZ CTE Programs of Study

8.1	Identify the purpose of version control systems (e.g., Git, Mercurial)				X
			Describe why Git would be considered a sharing platform Explain why tracking historical modifications are important on programming project	Git Mercurial Versions Control System	
8.2	Create a new repository				X
			Create a new repository	Repository	
8.3	Add source code or data to repository				X
			Given examples (scenario) Identify whether you would add, push, pull or restore	Push Pull	
8.4	Push changes to an upstream/master repository				X
			Given examples (scenario) Identify whether you would add, push, pull or restore	Upstream	
8.5	Pull changes from the upstream/master repository				
			Given examples (scenario) Identify whether you would add, push, pull or restore		
8.6	Restore previous versions of code from the repository				X
			Given examples (scenario) Identify whether you would add, push, pull or restore		

This project partially funded in Partnership with AZ CTE Programs of Study

Standard 9.0 - DEMONSTRATE PROGRAM ANALYSIS AND DESIGN					
Measurement Criteria		**Concepts**	**Implementation**	**Terminology**	**Testing Item**
9.1	Explain the steps in the System Development Life Cycle (SDLC) (e.g., planning, analysis, design, development, testing, implementation, maintenance)		List the steps in a program development cycle		x
9.2	Interpret a problem statement and identify program requirements		When given a problem, identify the program requirements		x
				Doubles Integers Strings Character	
9.3	Develop program requirements/specifications and a testing plan (e.g., user stories, automated testing, test procedures)				X
			Provide examples of user stories (needs and wants) Describe when it is appropriate to use automated testing in software development Describe the benefits of well-defined test procedures		
9.4	Determine input and output				X

This project partially funded in Partnership with AZ CTE Programs of Study

			Determine the correct data required from the user and by the user for successful execution of the program		
9.5	Choose appropriate data structures				X
				Integer Boolean Double Array ArrayList String Char	
9.6	Differentiate between bottom-up and top-down design				X
			Give examples of bottom up and top down design		
9.7	Apply pseudocode or graphical representations to plan the structure of a program or module (e.g., flowcharting, white boarding, UML)		Given a problem identify the pseudocode	Pseudocode Flowcharting UML	X
			Write and interpret pseudocode		
9.8	Choose appropriate documentation for a module				X
			Locate documentation for an existing class Write documentation at various levels		
9.9	Differentiate among major programming paradigms (e.g., imperative vs. declarative, object-oriented, functional)				X
			Explain the various types of programming	OOP- Object Oriented Programming	

This project partially funded in Partnership with AZ CTE Programs of Study

			Distinguish the difference between imperative and declarative programming paradigms	Functional/Procedural Imperative Declarative	

Standard 10.0 - DEVELOP A PROGRAM USING INDUSTRY BEST PRACTICES					
Measurement Criteria		**Concepts**	**Implementation**	**Terminology**	**Testing Item**
10.1	Use a program editor to enter and modify code		Determine what the best choices are for editing code Describe what text editor is used for		**X**
				DE-Integrated Development Environment SDK-Software Development Kit	
10.2	Differentiate between interpreted and compiled code (e.g., steps necessary to run executable code)				**X**
			Differentiate between interpreted and compiled code	Executable Code Interpreted Code Compiled Cole Machine Language Byte Code Assembly Language	
10.3	Identify the purpose of a build system (e.g., make, rake, ant, maven, SCons, grunt)				**X**
				Build Systems	

This project partially funded in Partnership with AZ CTE Programs of Study

10.4	Apply industry standards in documentation (e.g., self-documenting code; function-level, program-level, and user-level documentation)		Provide examples of in line documentation Explain the purpose of in line documentation Identify components that belong in program documentation List important standards in documentation Explain why reading comments and remarks is important for understanding a section coded in unfamiliar source code	Function Level Program Level User Level	**X**
10.5	Name identifiers and formatting code by applying recognized conventions		Determine dimensions given a problem Identify the correct variable name		**X**
			Determine the correct convention and purpose according to the language	Camelcase (Upper/Lower)	
10.6	Find and use program and language documentation				**X**
			Locate on line language libraries and identify documentation Given an example of an online library identify documentation		

This project partially funded in Partnership with AZ CTE Programs of Study

Standard 11.0 - TEST AND DEBUG TO VERIFY PROGRAM OPERATION					
Measurement Criteria		**Concepts**	**Implementation**	**Terminology**	**Testing Item**
11.1	Identify errors in program modules		Explain how one would identify syntax errors	Syntax error	**X**
			Interpret syntax error results	Compiler	
11.2	Identify boundary cases and generate appropriate test data		Identify test data when given a problem Identify boundaries given a problem		**X**
				Boundary Case	
11.3	Perform integration testing including tests within a program to protect execution from bad input or other runtime errors		Perform integration testing Given a scenario identify potential integration testing	Integration Testing	**X**
11.4	Categorize, identify, and correct errors in code, including syntax, semantic, logic, and runtime		Describe what to do if a program is producing unexpected numerical output Identify the items included when checking code	Semantic Error Logic Error Runtime Error Syntax Error	**X**
			Explain the hardest type of error to solve and why		
11.5	Perform different methods of debugging (e.g., hand-trace code or real time debugging tools)				**X**
			Identify errors generated by a loop by hand tracing the code	Real Time Debugging Hand-Trace Code	

Standard 12.0 - WRITE CODE TO PERFORM ARITHMETIC CALCULATIONS				
Measurement Criteria	**Concepts**	**Implementation**	**Terminology**	**Testing Item**

This project partially funded in Partnership with AZ CTE Programs of Study

12.1	Identify and correctly use arithmetic operations, applying the order of operations (precedence) with respect to programming		Identify proper formula to sum the value in variables given a problem		**X**
				Gemdas	
12.2	Interpret and construct mathematical formulas		Solve for incrementing variables		**X**
				Increment Decrement	
12.3	Use basic algorithms				**X**
			Use basic algorithms Write an algorithm	Algorithms	
12.4	Identify correct and problematic uses of integers, floating-point numbers, and fixed-point numbers in arithmetic				**X**
			Correctly define the format of the input and output specifications	Floating Point Fixed Point Numbers	

Standard 13.0 - EMPLOY MODULARITY IN WRITING PROGRAMS

Measurement Criteria		**Concepts**	**Implementation**	**Terminology**	**Testing Item**
13.1	Use standard library functions				**X**
			Incorporate standard library functions into existing code		

This project partially funded in Partnership with AZ CTE Programs of Study

13.2	Find and use third party libraries (e.g., on the web, package managers)				**X**
			Describe a third party library Explain the advantages and disadvantages of using third party libraries	Third Party Library	
13.3	Demonstrate refactoring techniques to reduce repetitious code and improve maintainability				**X**
			Given a code segment determine a more efficient way to accomplish the same results		
13.4	Demonstrate the use of parameters to pass data into program modules		Given a Math Library determine which method requires more than one parameter/argument	Parameter	**X**
				Argument	
13.5	Demonstrate the use of return values from modules		Solve the correct return value using Boolean isAlive ( )		**X**
			Solve correct return values using mathematical formulas Solve correct return values using String Operations		

Standard 14.0 - UTILIZE CONDITIONAL STRUCTURES IN WRITING PROGRAMS

Measurement Criteria		**Concepts**	**Implementation**	**Terminology**	**Testing Item**
14.1	Select correct syntax for decision statements (e.g., if/else, if, switch case)		Explain what an “if” statement is		**X**

This project partially funded in Partnership with AZ CTE Programs of Study

			Hand Trace loops Hand Trace conditional statements		
14.2	Compare values using relational operators (e.g., =, >, =, <=, not equal)		Identify < >= Explain what the Boolean conditional statement evaluates Given a code determine what range will cause normal value to be displayed Using the Boolean expression solve problems given		**X**
14.3	Evaluate Boolean expressions (e.g., AND, OR, NOT, NOR, XOR)		Using the Boolean expression solve problems given		**X**
				NOR XOR	
14.4	Select an appropriate decision structure for a given situation				**X**
			Write various decision structures using if, then, else		
14.5	Select the correct nesting for decision structures				**X**
			Explain and demonstrate a nested IF statement	Nested IF Statement	

Standard 15.0 -UTILIZE ITERATIVE STRUCTURES IN WRITING PROGRAMS				
Measurement Criteria	**Concepts**	**Implementation**	**Terminology**	**Testing Item**

This project partially funded in Partnership with AZ CTE Programs of Study

15.1	Identify various types of iteration structure (e.g., while, for, for-each, recursion)		Explain for and while loops Identify repetitive structures		**X**
			Explain the difference between for and four-each loop and when they should be used Identify the special cases where recursion is the most beneficial	Recursion Repetitive Iteration Loop	
15.2	Identify how loops are controlled (variable conditions and exits)		Given a problem identify which variable controls the number of times the loop iterates		**X**
			Describe all three parameters that controls a loop (Initial , Exit, Increment)		
15.3	Select the correct syntax for nested loops		Given a problem solve nested loops		**X**
			Given a problem solve nested loops by traversing and recording the value of each variable at each iteration	Traverse	
15.4	Compute the values of variables involved with nested loops		Given a problem solve nested loops		**X**
			Given a problem solve nested loops by traversing and recording the value of each variable at each iteration		
15.5					**X**

Standard 16.0 - UTILIZE PRIMITIVE DATA TYPES AND STRINGS

Measurement Criteria	Concepts	Implementation	Terminology	Testing Item

This project partially funded in Partnership with AZ CTE Programs of Study

16.1	Declare numeric, Boolean, character, and string variables		Explain why Boolean is a valid type variable Explain a string data type	Boolean	**X**
16.2	Choose the appropriate data type for a given situation		Identify the device that would result in a one-bit number Identify the variable used to store a whole number, a number with a decimal point, store a password etc. Explain what Float is used for	Float	**X**
			Explain the differences between float and double		
16.3	Identify the correct syntax for constants in a program		Explain constant variables		**X**
16.4	Identify the correct syntax for initializing and modifying variables				**X**
			Explain the importance of initializing a variable		
16.5	Identify the correct syntax and safe functions for operations on strings, including length, substring, and concatenation				**X**
			Use the correct syntax to ensure the computer gives correct results Use predefined procedures correctly	Concatenation	

This project partially funded in Partnership with AZ CTE Programs of Study

16.6	Explain complications of storing and manipulating dates				**X**
			Explain user requirements especially as they relate to different country's dates Define how a program manipulates dates Identify various date formats and how they need to be manipulated		

Standard 17.0 - UTILIZE BASIC DATA STRUCTURES IN PROGRAMS					
Measurement Criteria		**Concepts**	**Implementation**	**Terminology**	**Testing Item**
17.1	Demonstrate basic uses of arrays, including initialization, storage, and retrieval of values		Identify what type of information would use an array as a software choice Given a problem name the array and how many elements it contains		**X**
17.2	Distinguish between arrays and hashmaps (associative arrays)		Identify the best structure to store names and ages		**X**
			Give an example of when to use a hashmap for security purposes	Hashmaps	
17.3	Identify techniques for declaring, initializing, and modifying user-defined data types				**X**
			Give examples of programs that use user-defined data types		
17.4	Search and sort data in an array				**X**

This project partially funded in Partnership with AZ CTE Programs of Study

			Program a Bubble Sort, an Insertion Sort and a Merge Sort	Bubble Sort Insertion Sort Merge Sort	
17.5	Create and use two-dimensional arrays				**X**
			Create and use two-dimensional arrays	2-dementional array	
17.6	Describe the efficiency of different sorting algorithms (e.g., bubble, insertion, merge)		Explain bubble sort Explain insertion sort		
			Identify which sort to use in a given situation	Algahtythm	
17.7	Describe the efficiency of linear vs. binary searches [e.g., o(n) or o(log n)]				**X**
			Determine when to use a linear search versus a binary search	Linear search Binary Search	

Standard 18.0 - IDENTIFY WAYS TO INPUT AND OUTPUT INFORMATION

Measurement Criteria		**Concepts**	**Implementation**	**Terminology**	**Testing Item**
18.1	Identify correct input/output statements in a program		Identify samples of output reports		**X**
			Define user input requirements needed to produce correct output		
18.2	Choose the correct method of assigning input to variables, including data sanitization				**X**

This project partially funded in Partnership with AZ CTE Programs of Study

			Discuss the way data will be used (public, private, calculated) Use correct format of data		
18.3	Choose the correct method of outputting data, with formatting and escaping				**X**
			Determine user requirement and readability for output data		
18.4	Employ graphics methods to create images at specified locations				**X**
			Differentiate between screen coordinates and Cartesian	Screen Coordinates Cartesian Coordinates	
18.5	Choose correct GUI objects for input and output of data to the GUI interface (e.g., text boxes, labels, radio buttons, check boxes, dropdowns, list boxes)		Explain what a drop down box is	GUI Graphical User Interface Radio Button dropdowns,	**X**
			Use graphics to enhance user interfaces	text boxes labels check boxes list boxes	

Standard 19.0 - USE EXTERNAL DATA SOURCES WITHIN A PROGRAM					
Measurement Criteria		**Concepts**	**Implementation**	**Terminology**	**Testing Item**
19.1	Input data from a sequential file and database		Given a code segment determine the action taken (reads from file)		**X**

This project partially funded in Partnership with AZ CTE Programs of Study

			Write a program that uses an external sequential file	Sequential file	
19.2	Output data to a sequential file and/or database		Given a code segment determine the action taken (writes to a file)		**X**
19.3	Append data to an existing file		Given a code segment determine the action taken (appends to file)	Appends	**X**
19.4	Update files and/or databases		Given a code segment determine the action taken (updates a file)		**X**

Standard 20.0 - EMPLOY OBJECT-ORIENTED PROGRAMMING TECHNIQUES

Measurement Criteria		**Concepts**	**Implementation**	**Terminology**	**Testing Item**
20.1	Make a distinction between an object and a class		Explain how an OBJECT is different from a CLASS	OBJECT	**X**
20.2	Differentiate among inheritance, composition, and class relationships			Inheritance Encapsulation	**X**
			Explain the concepts of inheritance, composition and class relationships, and how they differ	Composition Inheritance Encapsulation	
20.3	Instantiate objects from existing classes				**X**
			Define the purpose of instantiation	Instantiate	

This project partially funded in Partnership with AZ CTE Programs of Study

			Write code that creates (instantiates) an object from both a standard class and a user-defined class		
20.4	Read the state of an object by invoking accessor methods				**X**
			Interpret and write code that reads an object (accessor method)	State Accessor method	
20.5	Change the state of an object by invoking a modifier method				**X**
			Interpret and write code that changes an object (modifier or "mutator" method); Given code segments, determine whether code is an accessor or a mutator method	Mutator method	
20.6	Determine the requirements for constructing new objects by reading the documentation				**X**
			Explain "creating an instance of an object"; Interpret and write constructor code (create an instance of an object)	Constructor Instance	
20.7	Create a user-defined class				**X**
			Explain the purpose of user-defined classes. Write code to create an object according to requirements from a user-defined class	User defined class	
20.8	Create a subclass of an existing class				**X**
			Explain the purpose of a subclass. How does that differ from a superclass. Write code that creates a subclass	Subclass Superclass	

Standard 21.0 - EMPLOY RUN-TIME ERROR-HANDLING TECHNIQUES

This project partially funded in Partnership with AZ CTE Programs of Study

Measurement Criteria		Concepts	Implementation	Terminology	Testing Item
21.1	Identify run-time errors		Explain how run-time errors occur	Run-time error	**X**
			Explain how a run-time error is different from a compile-time error. Identify if dividing by zero is a syntax, compile-time, or run-time error	Compile-time error Program bug Program execution	
21.2	Describe error-handling strategies		Explain key words "try" and "catch" Explain how to identify Run-time errors	Float	**X**
			Describe a "throw" statement and the "Throwable" class	Try Catch Finally Throw IO Exception	
21.3	Handle unexpected return values				**X**
			Given a code segment identify the type of value will be returned Identify if any a method having a return type set as "void" return	Incompatible Types Void Return Type Return	
21.4	Handle (catch) run-time errors and take appropriate action				**X**
			Given a code segment containing an undefined variable or a nonexistent method modify the code to handle the error with a try catch finally statement	Try Catch Finally Illegal Operation Exception	

This project partially funded in Partnership with AZ CTE Programs of Study

21.5	Throw standard exception classes		Explain the use of the word “throw”		**X**
			Demonstrate the use of standard exception classes and packages implicitly imported into classes	Throw Throwable Exception IOException	
21.6	Develop and throw custom exception classes				**X**
			Demonstrate writing your own exception class to handle code and classes that throw more than one related exception	Extends Super Throwable	

Terminology

2-dimensional array- a data structure that contain rows and columns of data. Each element position requires an index specifying the row and an index specifying the column.

A

Accessor method – returns the value of a private field.

Algorithmic Efficiency- algorithmic efficiency are the properties of an algorithm which relate to the amount of computational resources used by the algorithm. An algorithm must be analyzed to determine its resource usage

Algorithms- a process or set of rules to be followed in calculations or other problem-solving operations especially by a computer

This project partially funded in Partnership with AZ CTE Programs of Study

Anti-virus software- sometimes known as anti-malware software is computer software used to prevent detect and remove malicious software. Antivirus software was originally developed to detect and remove computer viruses hence the name

Appends- add (something) as an attachment or supplement

Array- an arrangement of items at equally spaced addresses in **computer** memory. **Array** data type used in a programming language to specify a variable that can be indexed. Associative **array** an abstract data structure model that generalizes **arrays** to arbitrary indices individually

ArrayList- a class which implements List interface. It is widely used because of the functionality and flexibility it offers. Most of the developers choose Arraylist over Array as it's a very good alternative of traditional java arrays.

Assembly- see assembly language below – refers to a lower level computer language

Assembly Language- a low-level symbolic code converted by an assembler

B

Big O notation- A theoretical measure of the execution of an algorithm usually the time or memory needed given the problem size n which is usually the number of items. Informally saying some equation f(n) = **O**(g(n)) means it is less than some constant multiple of g(n)

Binary – The core number system that most computers use to calculate and process logic

Binary Search - Binary search is an efficient algorithm for finding an item from an ordered list of items. It works by repeatedly dividing in half the portion of the list that could contain the item, until you've narrowed down the possible locations to just one.

Bitwise- designating an operator in a programming language that manipulates the individual bits in a byte or word

Boolean- denoting a system of algebraic notation used to represent logical propositions especially in computing and electronics

Boundary Case- a behavior of a system when one of its inputs is at or just beyond its maximum or minimum limits. It is frequently used when discussing software testing.

This project partially funded in Partnership with AZ CTE Programs of Study

Bubble Sort- sometimes referred to as sinking sort is a simple sorting algorithm that repeatedly steps through the list to be sorted compares each pair of adjacent items and swaps them if they are in the wrong order

Buffer Overflow- a buffer overflow or buffer overrun is an anomaly where a program while writing data to a buffer overruns the buffer's boundary and overwrites adjacent memory locations. This is a special case of the violation of memory safety

Build Systems – software tools designed to automate the process of program compilation. Their primary goal is to efficiently create an executable.

Byte Code- is computer object code that is processed by a program usually referred to as a virtual machine rather than by the "real" computer machine the hardware processor

C

Camelcase (Upper/Lower)- is the practice of writing compound words or phrases such that each word or abbreviation begins with a capital letter. Camel case may start with a capital or especially in programming languages with a lowercase letters

Carpel tunnel syndrome - a painful repetitive strain injury of the hand and fingers caused by compression of a major nerve where it passes over the carpal bones through a passage at the front of the wrist alongside the flexor tendons of the hand.

Cartesian Coordinates- numbers that indicate the location of a point relative to a fixed reference point (the origin) being its shortest (perpendicular) distances from two fixed axes (or three planes defined by three fixed axes) that intersect at right angles at the origin

Char – primitive data type for a single character.

Character - The `Character` class wraps a value of the primitive type `char` in an object. An object of type `Character` contains a single field whose type is `char`.

Check boxes – a GUI component that allows users to select and deselect items on an input screen.

This project partially funded in Partnership with AZ CTE Programs of Study

CISPA- Cyber Intelligence Sharing and Protection Act - is a proposed law in the United States which would allow for the sharing of Internet traffic information between the U.S. government and technology and manufacturing companies. The stated aim of the bill is to help the U.S. government investigate cyber threats and ensure the security of networks against cyberattacks

Compiled Code- a computer program (or a set of programs) that transforms source code written in a programming language (the source language) into another computer language (the target language) with the latter often having a binary form known as object code.

Compiler- a computer program (or a set of programs) that transforms source code written in a programming language (the source language) into another computer language (the target language) with the latter often having a binary form known as object code.

Composition- is the design technique to implement a "**has-a"** relationship in classes. It is achieved by using instance variables that refer to other objects. For example, a Person "has a" Job.

Concatenation- in the context of programming, is the operation of joining two strings together. The term"concatenation" literally means to merge two things together.

Constructor- a block of code, which runs when you use the "new" keyword in order to instantiate an object. It looks like a method, however it is not a method. Methods have return type but constructors don't have any return type.

COPPA - Children's Online Privacy Protection Act of 1998- applies to the online collection of personal information by persons or entities under U.S. jurisdiction from children under 13 years of age. It details what a website operator must include in a privacy policy when and how to seek verifiable consent from a parent or guardian and what responsibilities an operator has to protect children's privacy and safety online including restrictions on the marketing to those under 13. While children under 13 can legally give out personal information with their parents' permission many websites disallow underage children from using their services altogether due to the cost and work involved in the law compliance.

Copyleft- an arrangement whereby software or artistic work may be used modified and distributed freely on condition that anything derived from it is bound by the same condition

D

This project partially funded in Partnership with AZ CTE Programs of Study

Data Brokers- also called an information broker or information reseller is a business that collects personal information about consumers and sells that information to other organizations

Data Farms- the process of using a high performance computer or computing grid to run a simulation thousands or millions of times across a large parameter and value space. The result of data farming is a "landscape" of output that can be analyzed for trends anomalies and insights in multiple parameter dimensions

Data Integrity - a fundamental component of information security. In its broadest use "data integrity" refers to the accuracy and consistency of data stored in a database data warehouse data mart or other construct

Data Piracy- the stealing of legally protected software. Under copyright law software piracy occurs when copyright protected software is copied distributed modified or sold. Software piracy is considered direct copyright infringement when it denies copyright holders due compensation for use of their creative works.

Data Pricing – the process of buying and selling data online with respect to how much is charged for specific queries, and the nature of those queries.

Declarative- denoting high-level programming languages that can be used to solve problems without requiring the programmer to specify an exact procedure to be followed

Decrement- cause a discrete reduction in (a numerical quantity)

Digital Signature- a type of electronic signature that encrypts documents with digital codes that are particularly difficult to duplicate

DNA - molecule that carries genetic instructions in living organisms. Identified as an area of "future trends" in computing – see standard 7.5.

Doubles- a data type used to represent numbers with a decimal point (e.g., a real number or a floating-point number)

Dropdowns- a GUI interface that allows the user to select a single option from a list of options

E

This project partially funded in Partnership with AZ CTE Programs of Study

Encapsulation- the process of hiding data and including data within specific classes

Encryption- the most effective way to achieve data security. To read an encrypted file you must have access to a secret key or password that enables you to decrypt it. Unencrypted data is called plain text; encrypted data is referred to as cipher text

E-Trash/Waste- describes discarded electrical or electronic devices. Used electronics which are destined for reuse resale salvage recycling or disposal are also considered e-waste. Informal processing of electronic waste in developing countries may cause serious health and pollution problems as these countries have limited regulatory oversight of e-waste processing.

Executable Code- is used to describe sequences of executable instructions that do not necessarily constitute an executable file; for example sections within a program

F

FERPA- The Family Educational Rights and Privacy Act- is a Federal law that protects the privacy of student education records. The law applies to all schools that receive funds under an applicable program of the U.S. Department of Education.

Firewall- is a network security system that monitors and controls the incoming and outgoing network traffic based on predetermined security rules. A firewall typically establishes a barrier between a trusted secure internal network and another outside network such as the Internet that is assumed to not be secure or trusted

Fixed Point Numbers- are a simple and easy way to express fractional numbers using a fixed number of bits. Systems without floating-point hardware support frequently use fixed-point numbers to represent fractional numbers.

Float - A method for storing and calculating numbers in which the decimal points do not line up as in fixed point numbers.

Floating Point- the formulaic representation that approximates a real number so as to support a trade-off between range and precision.

Flowcharting- a tool for analyzing processes. It allows you to break any process down into individual events or activities and to display these in shorthand form showing the logical relationships between them. Constructing flowcharts promotes better understanding of processes and better understanding of processes is a pre-requisite for improvement.

This project partially funded in Partnership with AZ CTE Programs of Study

Function Level- refers to one of the two contrasting programming paradigms identified by John Backus in his work on programs as mathematical objects the other being value-level programming.

Functional/Procedural- types of programming languages. Procedural language is a set of instructions followed by the computer step-by-step. Functional language emphasize rules and pattern matching. Although functional code may appear non-intuitive, it provides succinct and natural programming structures for those who gain experience. Functional code is especially helpful in mathematics.

G

Gemdas- A math brain tip or trick used to remember the order (PEMDAS is not as accurate) of how to do numerical expressions.
1) G Grouping () l l {} or square root
2) E Exponent
3) MD or DM Multiplication & Division~left to right
4) AS or SA Addition & Subtraction~left to right

General Public License- a widely used free software license which guarantees end users (individuals organizations companies) the freedoms to run study share (copy) and modify the software

Git- a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency. Git is easy to learn and has a tiny footprint with lightning fast performance

GTD- Getting Things Done a time management method described in a book of the same title by productivity consultant David Allen

GUI (**Graphical User Interface)** – the visual layout of an application

H

Hacking - someone who seeks and exploits weaknesses in a computer system or computer network

This project partially funded in Partnership with AZ CTE Programs of Study

Hand-Trace Code- tracing involves a specialized use of logging to record information about a program's execution. This information is typically used by programmers for debugging purposes and additionally depending on the type and detail of information contained in a trace log by experienced system administrators or technical support personnel and software monitoring tools to diagnose common problems with software

Hashing -the transformation of a string of characters into a usually shorter fixed-length value or key that represents the original string. Hashing is used to index and retrieve items in a database because it is faster to find the item using the shorter hashed key than to find it using the original value. It is also used in many encryption algorithms

Hashmaps- (or hash table) a data structure used to implement an associative array, a structure that can **map** keys to values. A **hash table** uses a **hash** function to compute an index into an array of buckets or slots, from which the desired value can be found.

Higher-Level Language- a computer language that hides many of the complex details of the computer language from the programmer. It uses more natural language, making the code simpler and easier to understand. More user friendly than lower level languages.

HIPAA- the acronym for the Health Insurance Portability and Accountability Act that was passed by Congress in 1996 - HIPAA protects health insurance coverage for workers and their families when they change or lose their jobs

I

IDE-Integrated Development Environment- a software application that provides comprehensive facilities to computer programmers for software **development**. An **IDE** normally consists of a source code editor build automation tools and a debugger. Most modern **IDEs** have an intelligent code completion.

Identify Theft - the fraudulent acquisition and use of a person's private identifying information usually for financial gain

Imperative- a programming paradigm that describes computation in terms of statements that change a program state. Imperative programs define sequences of commands for the computer to perform. It contrasts with declarative programming which defines the logic and the outcome, but not the actual control flow.

This project partially funded in Partnership with AZ CTE Programs of Study

Increment- an increase or addition especially one of a series on a fixed scale

Industry-Related Data – information that is unique to its particular business sector

Inheritance – allows a subclass to use attributes and procedures defined in its parents class

Insertion Sort- an elementary sorting algorithm that sorts one element at a time. Most humans when sorting a deck of cards will use a strategy similar to insertion sort. The algorithm takes an element from the list and places it in the correct location in the list

Instance- a computational object bearing the attributes and behavior specified by a class.

Instantiate –the process of creating a new object or instance of a class.

Integers- the set of whole numbers and their opposites. Whole numbers greater than zero are called positive integers. Whole numbers less than zero are called negative integers. The integer zero is neither positive nor negative and has no sign

Integrated Circuits- sometimes called a chip or microchip is a semiconductor wafer on which thousands or millions of tiny resistors capacitors and transistors are fabricated. An IC can function as an amplifier oscillator timer counter computer memory or microprocessor

Integration Testing- the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing

Interpreted Code- programming language for which most of its implementations execute instructions directly without previously compiling a program into machine-language instructions. The interpreter executes the program directly translating each statement into a sequence of one or more subroutines already compiled into machine code.

Intrusion Detection- a device or software application that monitors network or system activities for malicious activities or policy violations and produces electronic reports to a management station

J

This project partially funded in Partnership with AZ CTE Programs of Study

JIT- Just in Time: an inventory strategy companies use to employ efficiency and decrease waste by receiving goods only as they are needed in the production process. Also used for training -- JIT training educates employees just as they need it so the training is used quickly with less time to forget the details

K

Kanban- an inventory control system developed by Taiichi Ohno an industrial engineer at Toyota as a system to improve and maintain a high level of production.

L

Labels- a GUI object that displays text, usually to describe the roles of other window objects.

Linear search- also known as sequential search is a process that checks every element in the list sequentially until the desired element is found. The computational complexity for linear search is O(n) making it generally much less efficient than binary search (O(log n))

List boxes- a box on the screen that contains a list of options only one of which can be selected

Logic Error- a bug in a program that causes it to operate incorrectly but not to terminate abnormally (or crash). A **logic error** produces unintended or undesired output or other behavior although it may not immediately be recognized as such

M

Machine Code- a computer programming language consisting of binary or hexadecimal instructions that a computer can respond to directly

Machine Language a computer programming language consisting of binary or hexadecimal instructions that a computer can respond to directly

Mercurial- One example of a free, cross-platform, revision control tool for software developers.

This project partially funded in Partnership with AZ CTE Programs of Study

Merge Sort- a recursive algorithm that continually splits a list in half. If the list is empty or has one item it is sorted by definition (the base case). If the list has more than one item we split the list and recursively invoke a merge sort on both halves

Microprocessor- an integrated circuit that contains all the functions of a central processing unit of a computer

Miniaturization- is the trend to manufacture ever smaller mechanical optical and electronic products and devices. Examples include miniaturization of mobile phones computers and vehicle engine downsizing.

Mutator method – or Modifier method – code that sets the value of a private field.

Modulo- (in number theory) with respect to or using a modulus of a specified number. Two numbers are congruent modulo a given number if they give the same remainder when divided by that number

O

OBJECT – Instance of class

OOP- Object Oriented Programming- refers to a type of computer programming (software design) in which programmers define not only the data type of a data structure but also the types of operations (functions) that can be applied to the data structure

Open Source- denoting software for which the original source code is made freely available and may be redistributed and modified

Optical- an optical computer is a device that uses the photons in visible light or infrared (IR) beams, rather than electric current, to perform digital computations. Optics is identified as an area of interest in future trends in computing – see Standard 7.5.

P

Parameter- describes data or variables passed into a method function or subroutine

Phishing - is the attempt to acquire sensitive information such as usernames passwords and credit card details (and sometimes indirectly money) often for malicious reasons by masquerading as a trustworthy entity in an electronic communication

This project partially funded in Partnership with AZ CTE Programs of Study

Program Change Request- a suggested enhancement or problem fix usually submitted on a form. Used to communicate the update to all parties; also serves as an audit trail for all programs/parts affected. Also known as an "engineering change request" (ECR) or a "work order"

Program Level Documentation- explanation of methods, parameters, and code, as it pertains to individual programs; as opposed to system documentation, sequences of interactions between users, or static class dependencies.

Pseudocode- artificial informal computer language

Pull- To request data from another program or computer

Push- where data is sent without a request being made

Q

Quantum - the smallest discrete quantity of some physical property that a system can possess. Identified as an area of interest in future trends in computing – see Standard 7.5.

R

Radio Button – the GUI component that allows the user to only chose one option from a list of options

Real Time Debugging- the process of finding and resolving bugs or defects that prevent correct operation of computer software or a system. Debugging tends to be harder when various subsystems are tightly coupled as changes in one may cause bugs to emerge in another

Repository - a central location in which data is stored and managed

Runtime Error- An error that occurs during the execution of a program. In contrast compile-time errors occur while a program is being compiled. Runtime errors indicate bugs in the program or problems that the designers had anticipated but could do nothing about. For example running out of memory will often cause a runtime error

This project partially funded in Partnership with AZ CTE Programs of Study

S

Sanitizing- to erase a storage device such as a computer hard drive so thoroughly that no residual data can be collected from the device

Satellite- A computer located remotely from the host computer or under the control of the host. It can function as a slave to the master computer or perform offline tasks

Screen Coordinates- similar to Cartesian Coordinates, except that origin is in top left corner of screen. Screen position is defined by X and Y points relative to the origin.

SDK-Software Development Kit- typically a set of software development tools that allows the creation of applications for a certain software package software framework hardware platform computer system video game console operating system or similar development platform

Semantic Error- Writing invalid program logic that produces incorrect results when the instructions are executed. The syntax of the source code may be valid but the algorithm being employed is not

Sequential file- sequential access means that a group of elements (such as data in a memory array or a disk file or on magnetic tape data storage) is accessed in a predetermined ordered sequence. Sequential access is sometimes the only way of accessing the data for example if it is on a tape.

Social Engineering- is the art of manipulating people so they give up confidential information.

Social Networking- the use of dedicated websites and applications to interact with other users or to find people with similar interests to oneself.

Software Cracking- is the modification of software to remove or disable features which are considered undesirable by the person cracking the software especially copy protection features (including protection against the manipulation of software serial number hardware key date checks and disc check) or software annoyances like nag screens and adware

Software license- a legally binding agreement that specifies the terms of use for an application and defines the rights of the software producer and of the end-user. All software must be legally licensed before it may be installed

This project partially funded in Partnership with AZ CTE Programs of Study

Spam- usually considered to be electronic junk mail or junk newsgroup postings. Some people define spam even more generally as any unsolicited email. However if a long-lost brother finds your email address and sends you a message this could hardly be called spam even though it is unsolicited

Spoofing- A technique used to gain unauthorized access to computers whereby the intruder sends messages to a computer with an IP address indicating that the message is coming from a trusted host

SQL -- Structured Query Language - the process of creating data-driven inquiries to extract data from a database

SQL Injection- SQL injection: insertion of data into a query to extract data from a database in a malicious manner to extract sensitive or confidential data

State – the set of all values of the variables of a program at any point during its execution.

String- traditionally a sequence of characters either as a literal constant or as some kind of variable. The latter may allow its elements to be mutated and the length changed or it may be fixed (after creation)

Subclass – a subclass is a class definition that inherits generalized behaviors from its parent, but then can also add specific behaviors of its own. It is characterized by the "is a" phrase: for example, banana "is a" fruit

Superclass – a superclass gives methods or behaviors to a subclass.

Syntax Error- a character or string incorrectly placed in a command or instruction that causes a failure in execution

T

Text boxes- a GUI interface that allows the user to type in a small amount of text.

Third Party Libraries- a reusable software component developed to be either freely distributed or sold by an entity other than the original vendor of the development platform. The third-party software component market thrives because many programmers believe that component-

This project partially funded in Partnership with AZ CTE Programs of Study

oriented development improves the efficiency and the quality of developing custom applications. Common third-party software includes macros bots and software/scripts to be run as add-ons for popular developing software

Transistors- a semiconductor device used to amplify or switch electronic signals and electrical power. It is composed of semiconductor material with at least three terminals for connection to an external circuit

U

UML- The Unified Modeling Language - OMG's most-used specification and the way the world models not only application structure behavior and architecture but also business process and data structure

Unethical- not morally correct

Upstream- upstream refers to the direction in which data can be transferred from the client to the server (uploading). This differs greatly from downstream not only in theory and usage but also in that upstream speeds are usually at a premium

User defined class- In the real world, we have much more complicated objects, so primitive data types are not sufficient. Object oriented programming allows us to model real-world objects. User defined classes combine the data and methods that operate on that data.

User input validation: the process of ensuring that a program operates on clean correct and useful data. Programs should contain rules constraints or "check routines" that check for correctness meaningfulness and security of data that are input to the system.

User Level- The protection of information and information systems against unauthorized access or modification of information whether in storage processing or transit and against denial of service to authorized users.

User-defined data types- data types that model the structure and behavior of the **data** in their applications.

V

This project partially funded in Partnership with AZ CTE Programs of Study

Vacuum Tubes- a glass tube that has its gas removed which creates a vacuum. Vacuum tubes contain electrodes for controlling electron flow in early computers that used them as a switch or an amplifier.

Versions Control System- A component of software configuration management version control also known as revision control or source control. Version Control manages changes to documents, computer programs, large web sites and other collections of information. Changes are usually identified by a number or letter code termed the "revision number" "revision level" or simply "revision." For example an initial set of files is "revision 1." When the first change is made the resulting set is "revision 2" and so on. Each revision is associated with a timestamp and the person making the change. Revisions can be compared, restored and with some types of files, merged.

Viruses- is a malware program that when executed replicates by inserting copies of itself (possibly modified) into other computer programs data files or the boot sector of the hard drive; when this replication succeeds the affected areas are then said to be "infected"

Voice Recognition- is the ability of a machine or program to identify words and phrases in spoken language and convert them to a machine-readable format. Rudimentary speech recognition software has a limited vocabulary of words and phrases and may only identify these if they are spoken very clearly

W

Waterfall- a sequential design process used in software development processes in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of conception initiation analysis design construction testing production/implementation and maintenance

White Collar Hacking -
a computer hacker who is hired by an organization to undertake nonmalicious hacking work in order todiscover computer-security flaws

Work order- a suggested enhancement or problem fix usually submitted on a form. Used to communicate the update to all parties; also serves as an audit trail for all programs/parts affected. Also referred to as a “program change request.”

This project partially funded in Partnership with AZ CTE Programs of Study

This project partially funded in Partnership with AZ CTE Programs of Study