FREEHOLD REGIONAL HIGH SCHOOL DISTRICT

OFFICE OF CURRICULUM AND INSTRUCTION

COMPUTER SCIENCE ACADEMY

HONORS COMPUTER SCIENCE I

Grade Level: 9

Credits: 5

Course Code: 296050

BOARD OF EDUCATION ADOPTION DATE:

AUGUST 31, 2015

FREEHOLD REGIONAL HIGH SCHOOL DISTRICT

Board of Education

Mr. Heshy Moses, President
Mrs. Jennifer Sutera, Vice President
Mr. Vincent Accettola
Mr. William Bruno
Mrs. Elizabeth Canario
Mr. Samuel Carollo
Mrs. Amy Fankhauser
Mrs. Kathie Lavin
Mr. Michael Messinger

Central Administration

Mr. Charles Sampson, Superintendent
Dr. Nicole Hazel, Chief Academic Officer
Dr. Jeffrey Moore, Director of Curriculum and Instruction
Ms. Stephanie Mechmann, Administrative Supervisor of Curriculum & Instruction
Dr. Nicole Santora, Administrative Supervisor of Curriculum & Instruction

Curriculum Writing Committee

Mr. James Gill
Ms. Rhonda Solomon

Supervisor

Mr. Joseph Iacullo

296050: HONORS COMPUTER SCIENCE I

COURSE PHILOSOPHY

In *Honors Computer Science I*, students will enhance their critical thinking and problem-solving skills and learn the fundamentals of computer programming in a fast-paced and rigorous environment. Through the creation of user-friendly programs that model and solve real-world scenarios and problems, students will develop skills in computer programming that are the foundation for understanding all programming languages. A focus of this course is to challenge the novice programmer to create effective, robust computer programs.

COURSE DESCRIPTION

Honors Computer Science I is the first computer science course in the Computer Science Academy. It provides the foundational knowledge and skills that students will need to be successful in future programming courses. Students will learn the basic structures of programming such as repetition statements, selection statements, arrays, classes, and recursion. They will learn these basics through the creation of user-friendly and efficient programs using the Java programming language.

COURSE SUMMARY

COURSE GOALS

CG1: Students will design solutions to complex problems by developing algorithms.
CG2: Students will apply their learning of programming concepts to decipher other programming languages.
CG3: Students will create programs that adhere to complex specifications.
CG4: Students will effectively design solutions to complex tasks by modularizing them into smaller ones.

COURSE ENDURING UNDERSTANDINGS	COURSE ESSENTIAL QUESTIONS
CEU1: There are multiple avenues to design solutions to problems and some may be more efficient than others.	CEQ1a: Can there be more than one algorithm to solve a problem and how can you determine whether one is better than another? CEQ1b: Can a program be effective while not efficient? CEQ1c: What is the meaning of “robust” and “elegant” in the world of programming?
CEU2: There are similarities and differences between all computer programming languages.	CEQ2: What are the essential similarities between programming languages? How does this help us to learn a language? CEQ2b: What are the similarities between programming and spoken languages?
CEU3: Writing a robust computer program requires the consideration of the specifications as well as predicting for unforeseen events.	CEQ3: What determines the use of decision making in program design?
CEU4: Writing efficient programs require that a problem be separated into independent modules.	CEQ4a: How do you go about solving a problem? CEQ4b: What makes a program maintainable?

UNIT GOALS & PACING		
UNIT TITLE	**UNIT GOALS**	**RECOMMENDED DURATION**
Unit 1: Introduction to Computer Programming	Students will effectively and efficiently create interactive programs by using algorithms and manipulating data with the use of variables.	4-6 weeks
Unit 2: Selection and Repetition Statements	Students will effectively and efficiently create interactive programs by incorporating selection and repetition statements.	5-7 weeks
Unit 3: Arrays, Methods, and Sorting	Students will effectively and efficiently incorporate arrays and write methods that will enhance the development of sorting and interactive programs.	5-7 weeks
Unit 4: Classes and Interfaces	Students will build and develop interfaces and classes to perform specific services for targeted purposes.	4-6 weeks
Unit 5: Enhancing Classes with Inheritance and Polymorphism	Students will create classes from pre-existing classes by using inheritance and polymorphism.	5-7 weeks
Unit 6: Recursion	Students will effectively and elegantly incorporate recursion when creating interactive programs.	5-7 weeks

296050: HONORS COMPUTER SCIENCE I
UNIT 1: Introduction to Computer Programming **SUGGESTED DURATION: 4-6 weeks**

UNIT OVERVIEW	
UNIT LEARNING GOALS	
Students will effectively and efficiently create interactive programs by using algorithms and manipulating data with the use of variables.	
UNIT LEARNING SCALE	
4	In addition to score 3 performances, the student can develop programs from more complex algorithms.
3	The student can: • describe the relationship between hardware and software; • describe the steps involved in program compilation and execution; • write an algorithm that is an ordered sequence of instructions for solving a problem; • write an algorithm when presented with a problem; • define the difference between primitive data and objects; • declare and use variables; • write programs using operations involving primitive data and the use of an input class.
2	The student sometimes needs assistance from a teacher, makes minor mistakes, and/or can do the majority of level 3 performances.
1	The student needs assistance in attempting to reach level 3.
0	Even with help, the student does not exhibit understanding of performances listed in level 3.

ENDURING UNDERSTANDINGS	ESSENTIAL QUESTIONS
EU1: An algorithm helps programmers formulate a plan to consider specifications as well as predict for unforeseen events.	EQ1: Could you efficiently write a program without first creating an algorithm?
EU2: A computer executes instructions exactly as written, not necessarily as intended.	EQ2: How do you confirm that what is written is what was intended by the programmer?
EU3: Determining data types when programming is integral to a program's effectiveness.	EQ3: How do you decide on what data type to use?
EU4: Programs that interact with users must consider the various possible human responses.	EQ4a: What if the user enters incorrect input? EQ4b: Why use live data?

NJCCCS & COMMON CORE STANDARDS

HSN-Q.A.2 Define appropriate quantities for the purpose of descriptive modeling.

SMP1 Make sense of problems and persevere in solving them.
SMP4 Model with mathematics.
SMP6 Attend to precision.
SMP7 Look for and make of structure.
SMP8 Look for and express regularity in repeated reasoning.

COMMON ASSESSMENT	
ALIGNMENT	DESCRIPTION
LG 1 EU1,EQ1 EU2,EQ2 EU3, EQ3 EU4,EQ4a,EQ4b HSN-Q.A.2 SMP 1, 4, 6, 7, 8 DOK 2	Quadratic Formula: Students will create a program that finds the roots of a quadratic equation when provided with the coefficients. Before creating the program, the students will have to create the algorithm for the program. Students must also review and be able to prove that their algorithm and their program take into consideration all possible coefficients.

SUGGESTED STRATEGIES		
ACTIVITIES	DECLARATIVE KNOWLEDGE	PROCEDURAL KNOWLEDGE
Students will hand write an algorithm of their choice (i.e., how to tie a tie). Student will select another student to actually follow the instructions written to determine whether the algorithm was written correctly.  The level of expectation for individual student's algorithm will be based on their ability.	parts of a computer system algorithm	Create an algorithm from a problem set DOK 3
Students will be directed to write code utilizing print statements that will generate various output.  The level of expectation for individual student's program will be based on their ability.	print statements variables	Develop a program from a simple algorithm and output certain information DOK 2

296050: HONORS COMPUTER SCIENCE I

UNIT 2: Selection and Repetition Statements **SUGGESTED DURATION: 5-7 weeks**

UNIT OVERVIEW

UNIT LEARNING GOALS

Students will effectively and efficiently create interactive programs by incorporating selection and repetition statements.

UNIT LEARNING SCALE

4	In addition to score 3 performances, the student can: • use more advanced conditional statements; • optimize nested if-statements to conjunctive Boolean statements resulting in more concise code.
3	The student can: • effectively use if/else statements; • use Boolean expressions when writing conditional statements; • appropriately incorporate loops in program design; • effectively use while statements; • effectively use for statements; • alter the flow of control of a program through the use of selection statements.
2	The student sometimes needs assistance from a teacher, makes minor mistakes, and/or can do the majority of level 3 performances.
1	The student needs assistance in attempting to reach level 3.
0	Even with help, the student does not exhibit understanding of performances listed in level 3.

ENDURING UNDERSTANDINGS	ESSENTIAL QUESTIONS
EU1: Program development can be enhanced by altering the flow of control via the use of conditional statements.	EQ1a: How do I choose which selection structure should be used to best facilitate the program? EQ1b: How do I determine when it is appropriate to execute statements out of sequence?
EU2: There are multiple ways to design and implement repeated operations.	EQ2: Is one method of looping more efficient or effective than another?

NJCCCS & COMMON CORE STANDARDS

HSN-Q.A.2 Define appropriate quantities for the purpose of descriptive modeling.

SMP1 Make sense of problems and persevere in solving them.
SMP4 Model with mathematics.
SMP6 Attend to precision.
SMP7 Look for and make of structure.
SMP8 Look for and express regularity in repeated reasoning.

COMMON ASSESSMENT	
ALIGNMENT	DESCRIPTION
LG 1 EU1,EQ1a,EQ1b EU2,EQ2 EU3, EQ3 EU4,EQ4a,EQ4b HSN-Q.A.2 SMP 1, 4, 6, 7, 8 DOK 3	Sum It Up: Students will design and implement an application that reads an integer value and prints the sum of all the even integers between 2 and the input value, inclusive. The program will print an error message if the input value is less than 2. It will also prompt the user accordingly. Students will have to justify which type of loop they chose and why.

SUGGESTED STRATEGIES		
ACTIVITIES	DECLARATIVE KNOWLEDGE	PROCEDURAL KNOWLEDGE
Students will complete truth tables applying the principles of Boolean logic.	Boolean values logical operators	Create a truth table and evaluate expressions using logical operators DOK 3
Students will be asked to create a program that deliberately includes an endless loop.  The level of expectation for individual student's program will be based on their ability.	for loop while loop	Create programs using different kinds of loops DOK 3

296050: HONORS COMPUTER SCIENCE I

UNIT 3: Arrays, Methods, and Sorting **SUGGESTED DURATION: 5-7 weeks**

UNIT OVERVIEW

UNIT LEARNING GOALS

Students will effectively and efficiently incorporate arrays and write methods that will enhance the development of sorting and interactive programs.

UNIT LEARNING SCALE

4	In addition to score 3 performances, the student can use the index of an array to organize and store data.
3	The student can: • write programs that use arrays to represent many data elements; • write programs and methods to modularize larger, more complex code; • write programs to sort data using iterative sorting methods including insertion and selection sorts; • write programs to perform a sequential search; • write programs to perform a binary search on a sorted array; • use parallel arrays, using one array to maintain the status of the second array.
2	The student sometimes needs assistance from a teacher, makes minor mistakes, and/or can do the majority of level 3 performances.
1	The student needs assistance in attempting to reach level 3.
0	Even with help, the student does not exhibit understanding of performances listed in level 3.

ENDURING UNDERSTANDINGS	ESSENTIAL QUESTIONS
EU1: An array is a simple but powerful way to group and organize data that holds a fixed number of values of the same type.	EQ1: What requirements would insinuate that arrays would be beneficial in a program?
EU2: A method is an optimal way to compartmentalize parts of a program.	EQ2: What are the advantages of compartmentalization?
EU3: There are many different methods of sorting; some may be more effective or efficient than others depending on the data.	EQ3: What criteria would you use to determine which sort is the most effective?

NJCCCS & COMMON CORE STANDARDS

HSN-Q.A.2 Define appropriate quantities for the purpose of descriptive modeling.
HSN.VM.C.6 Use matrices to represent and manipulate data.

SMP1 Make sense of problems and persevere in solving them.
SMP4 Model with mathematics.
SMP6 Attend to precision.
SMP7 Look for and make of structure.
SMP8 Look for and express regularity in repeated reasoning.

COMMON ASSESSMENT	
ALIGNMENT	DESCRIPTION
LG 1 EU1,EQ1 EU2,EQ2 EU3, EQ3 HSN-Q.A.2 HSN.VM.C.6 SMP 1, 4, 6, 7, 8 DOK 3	Student Grade Retrieval: Students will create a program that uses parallel arrays to maintain student names and corresponding grades in a specific course. The program will search for a name and print out the name and corresponding grade. The students will need to be able to explain why they must use a sequential search as opposed to a binary search to successfully code this program
LG 1 EU1,EQ1 EU2,EQ2 EU3, EQ3 HSN-Q.A.2 HSN.VM.C.6 SMP 1, 4, 6, 7, 8 DOK 3	Teachers' EZ Grader: Students will write a program that asks the user to enter in 10 grades. The grades will be stored in an array. The array of grades will be sent to a method that will sort the array in descending order. A second method will be called that calculates the average grade. The driver program will be responsible to print out the sorted array and the average grade. Students will have to justify which type of sort they used and why.

SUGGESTED STRATEGIES		
ACTIVITIES	DECLARATIVE KNOWLEDGE	PROCEDURAL KNOWLEDGE
Students will participate in a human sort. They will stand in line in random height order, and by using the selection sort algorithm they will sort themselves from shortest to tallest.	selection sort	Use the selection sort DOK 3
Students will code and execute a selection sort and determine the number of comparisons made.  The level of expectation for individual student's program will be based on their ability.	selection sort	Create a program that involves selection sort DOK 3
Students will play a game with a partner. The partner will choose a number from 1-100. Using the binary search, the student must guess the number in the smallest amount of guesses possible	binary search	Use the binary search DOK 3

296050: HONORS COMPUTER SCIENCE I

UNIT 4: Classes and Interfaces **SUGGESTED DURATION: 4-6 weeks**

UNIT OVERVIEW

UNIT LEARNING GOALS

Students will build and develop interfaces and classes to perform specific services for targeted purposes.

UNIT LEARNING SCALE

4	In addition to score 3 performances, the student can write programs that implement multiple interfaces.
3	The student can: • define classes that act like blueprints for new objects, made up of variables and methods; • explain encapsulation and Java modifiers; • explain and use method overloading; • divide complicated methods into simpler supporting methods; • describe relationships between objects.
2	The student sometimes needs assistance from a teacher, makes minor mistakes, and/or can do the majority of level 3 performances.
1	The student needs assistance in order to reach the learning goal.
0	Even with help, the student does not exhibit understanding of performances listed in level 3.

ENDURING UNDERSTANDINGS	ESSENTIAL QUESTIONS
EU1: A class is a blueprint from which individual objects are created.	EQ1: How is a class designed to incorporate all of the characteristics and behaviors of an object?
EU2: An interface is a collection of constants and abstract methods that enforce the design of the program to comply with specified requirements.	EQ2a: How do I know when to make a method abstract? EQ2b: Why use interfaces? What are the advantages?

NJCCCS & COMMON CORE STANDARDS

HSN-Q.A.2 Define appropriate quantities for the purpose of descriptive modeling.

SMP1 Make sense of problems and persevere in solving them.
SMP4 Model with mathematics.
SMP6 Attend to precision.
SMP7 Look for and make of structure.
SMP8 Look for and express regularity in repeated reasoning.

COMMON ASSESSMENT

ALIGNMENT	DESCRIPTION
LG 1 EU1,EQ1 EU2,EQ2a, EQ2b HSN-Q.A.2 SMP 1, 4, 6, 7, 8	Students will create an employee class that implements the Comparable interface. The instance variables will be first name, last name, employee number, and salary. Students will compare the employees by salary. Students will create an employee array and will pass it to a method to be sorted and printed. The program will include overloaded pay methods and method execution will be determined by parameters passed. Students will be asked to explain how visibility modifiers were used to ensure object encapsulation.

SUGGESTED STRATEGIES		
ACTIVITIES	DECLARATIVE KNOWLEDGE	PROCEDURAL KNOWLEDGE
Using the analogy of a blueprint, students will be asked to come up with real-life examples of objects that could be created.	blueprint objects	Develop objects using real-life examples DOK 3
Students will be asked to take a method from a previously written program, and to write two overloaded versions of that method.  The level of expectation for individual student's method will be based on their ability.	methods overloaded methods parameters	Develop overloaded methods from previously written methods DOK 3

296050: HONORS COMPUTER SCIENCE I

UNIT 5: Enhancing Classes with Inheritance and Polymorphism **SUGGESTED DURATION: 5-7 weeks**

UNIT OVERVIEW	
UNIT LEARNING GOALS	
Students will create classes from pre-existing classes by using inheritance and polymorphism.	
UNIT LEARNING SCALE	
4	In addition to score 3 performances, the student can develop class hierarchies combined with abstract classes and/or interfaces.
3	The student can: • derive new classes from existing ones; • explain how inheritance supports software reuse; • add and modify methods in child classes; • design class hierarchies; • define polymorphism and how it can be done.
2	The student sometimes needs assistance from a teacher, makes minor mistakes, and/or can do the majority of level 3 performances.
1	The student needs assistance in order to reach the learning goal.
0	Even with help, the student does not exhibit understanding of performances listed in level 3.

ENDURING UNDERSTANDINGS	ESSENTIAL QUESTIONS
EU1: Deriving new classes from existing classes via inheritance supports software reuse. By using existing software the programmer is reusing the design, the implementation, and the testing previously completed on existing software.	EQ1: How do we design a class hierarchy so that it can be reused in the future?
EU2: Every derivation should be an "is-a" relationship. The child should be a more specific version of the parent.	EQ2a: Why do we need to hide variables and methods? Why can't everything be visible? How do we determine what gets hidden and what doesn't?
EU3: A polymorphic reference promotes program robustness by permitting the flexibility of using the same method or variable names to perform similar functions but with different implementations to be determined upon program execution.	EQ3: How does the use of polymorphic references enhance the program design?

NJCCCS & COMMON CORE STANDARDS
HSN-Q.A.2 Define appropriate quantities for the purpose of descriptive modeling. SMP1 Make sense of problems and persevere in solving them. SMP4 Model with mathematics. SMP6 Attend to precision. SMP7 Look for and make of structure. SMP8 Look for and express regularity in repeated reasoning.

COMMON ASSESSMENT	
ALIGNMENT	DESCRIPTION
LG1 EU1,EQ1 EU3,EQ3 HSN-Q.A.2 SMP 1, 4, 6, 7, 8 DOK 3	Students will create a class hierarchy with the base class Pet and several child classes (i.e., Dog, Cat, Snake, etc.). Child classes will include methods that are Pet specific. Students will create a driver class that instantiates multiple Pet objects. The driver class will invoke polymorphic methods that represent behaviors such as communication and movement. Students will explain how inheritance is utilized in this example and the advantages of including the polymorphic methods in the driver class.

SUGGESTED STRATEGIES		
ACTIVITIES	DECLARATIVE KNOWLEDGE	PROCEDURAL KNOWLEDGE
Students will draw an inheritance hierarchy containing classes that represent different types of clocks. Students will show the variables and method names for two of these classes.	inheritance classes instance variables	Develop a simple inheritance structure DOK 3
Students will draw a class hierarchy for types of teachers at a high school. Students will show what characteristics would be represented in the various classes of the hierarchy. Students will explain how polymorphism could play a role in assigning courses to each teacher.  The level of expectation for individual student's program will be based on their ability.	inheritance classes polymorphic methods	Develop a class structure that uses polymorphic methods DOK 4

296050: HONORS COMPUTER SCIENCE I	COURSE NUMBER: COURSE NAME
UNIT 6: Recursion	SUGGESTED DURATION: 5-7 weeks

UNIT OVERVIEW	
UNIT LEARNING GOALS	
Students will effectively and elegantly incorporate recursion when creating interactive programs.	
UNIT LEARNING SCALE	
4	In addition to score 3 performances, the student can program a classic, challenging program using recursion, such as the *Towers of Hanoi.*
3	The student can: • explain the underlying idea of recursion; • examine recursive methods and processing steps; • define infinite recursion and discuss ways to avoid it; • explain when recursion should and should not be used; • demonstrate the use of recursion to solve problems.
2	The student sometimes needs assistance from a teacher, makes minor mistakes, and/or can do the majority of level 3 performances.
1	The student needs assistance in order to reach the learning goal.
0	Even with help, the student does not exhibit understanding of performances listed in level 3.

ENDURING UNDERSTANDINGS	ESSENTIAL QUESTIONS
EU1: Recursion, a programming technique in which a method calls itself, is used to simplify an iterative approach by performing simple function calls.	EQ1: How do you assess the advantage of incorporating a recursive solution versus an iterative solution when creating a program?

HSN-Q.A.2 Define appropriate quantities for the purpose of descriptive modeling.

SMP1 Make sense of problems and persevere in solving them.
SMP4 Model with mathematics.
SMP6 Attend to precision.
SMP7 Look for and make of structure.
SMP8 Look for and express regularity in repeated reasoning.

COMMON ASSESSMENT	
ALIGNMENT	DESCRIPTION
LG 1 EU1, EQ1 HSN-Q.A.2 SMP 1, 4, 6, 7, 8 DOK 3	Students will write two programs that will add up the values between 1 and *n* inclusive, where *n* is any positive number. The first program will complete the task by using iterative methods while the second program will incorporate recursion. Program algorithms will be compared to determine whether the differences reflect efficiency or programming elegance. Students will be asked to determine whether recursion is the best choice for writing the program and to justify their conclusion.

SUGGESTED STRATEGIES		
ACTIVITIES	DECLARATIVE KNOWLEDGE	PROCEDURAL KNOWLEDGE
Students will write a recursive program that will return a Boolean determining whether a String is a palindrome or not.	recursion Boolean values string palindrome	Create a recursive program that compares the beginning and end of a string to determine the structure of the string DOK 4
Students will write a recursive program that will utilize a binary search to locate and return the index of an element in a sorted array or a -1 if the element is not found.	recursion sorted array index	Develop a binary search on a sorted array returning an integer result DOK 4