

Summit Public Schools
Summit, New Jersey
Grade Level: 11-12 / Content Area: Mathematics

Course Overview:

Design Projects in Computer Science is a one semester course where students will have the opportunity to experience the Software Development Life Cycle (SDLC) from beginning to end. Students may choose to enroll in this course for multiple semesters. In previous courses students have learned many different programming concepts, and they will be able to put them all together and collaborate with each other to build programs from scratch.

The SDLC consists of five stages: Analyzing, Designing, Coding, Debugging and Testing, and Maintaining. Students will work through each stage of the cycle, focusing on planning and documenting their code as they go. Students will collaborate with each other to find faults in the design stage, bugs in the coding and testing stage, and to help plan updates to the program during the maintenance stage.

This course will help prepare students to work on programs as part of a team in a real world environment, and write programs that are well planned and easy for an outside reader to understand.

Prerequisite Course:

AP Computer Science A

Suggested Timeline:

Since students will be coming up with their own plans, not every program will take the same amount of time to complete. Some students may end up completing one program per marking period, while others may complete more. The timeline is written for a student who will complete one project per marking period. Each time students move through the SDLC it will reinforce the ideas they have learned the last time they complete a project. Students should complete at least two projects during a semester.

Each of the following stages could have within it its own cycle of the 5 stages if any modifications need to be made to the original plan. This is a natural occurrence in the SDLC.

Analyzing	4 Days
Designing	8 Days
Coding and Debugging	7 Days
Testing	6 Days
Maintaining	5 Days

Stage 1: Analyzing	
Big Ideas: <i>Course Objectives/ Content Statement(s)</i> Students will understand... • The role that Analyzing plays in the Software Development Life Cycle • How to think of a program as a set of inputs and outputs • How to come up with an idea that can be executed in a reasonable amount of time	
Essential Questions <i>What provocative questions will foster inquiry, understanding, and transfer of learning?</i>	Enduring Understandings <i>What will students understand about the big ideas?</i>
• What type of program do I want to make? • How can the program be broken up into inputs and outputs? • What outside information will I need to come up with a design? • What should be included in the first stage of my program?	Students will understand that: • Analyzing involves coming up with an idea without writing any code yet • As a professional programmer, this stage could involve working with a client to come up with a product idea that satisfies the clients needs • Programs have a life cycle, this is not the last chance to analyze and plan, just the first. • Planning one step at a time is a key aspect of writing effective programs that are easy to work with, and easy to read.
Areas of Focus: Proficiencies (New Jersey Student Learning Standards)	Activities
Students will: • Come up with ideas for a program, and choose one to implement • Create a set of inputs and outputs for a program • Create an outline of what the program will look like and how it will function from a user perspective NJ Computer Science and Design Thinking Standards • 8.1.12.CS.3 Successful troubleshooting of complex problems involves multiple approaches including research, analysis, reflection, interaction with peers, and drawing on past experiences. • 8.1.12.IC.1 The design and use of computing technologies and artifacts can positively or	During this stage of the SDLC students will come up with one or more idea(s) for a program, and choose one to implement based on timing and interest. The purpose of this stage is to think about what the program will look like, and how it will function without coding. Students should focus on input and output when they are analyzing since that is all the user will see when running the program. This could involve a list, or diagrams or both of what the program will look like to the user. Students will submit their plan before moving on to the next stage in the SDLC

negatively affect equitable access to information and opportunities.

- **8.1.12.DA.1** Individuals select digital tools and design automated processes to collect, transform, generalize, simplify, and present large data sets in different ways to influence how other people interpret and understand the underlying information.
- **8.1.12.DA.2** Choices individuals make about how and where data is organized and stored affects cost, speed, reliability, accessibility, privacy, and integrity.
- **8.1.12.DA.3** Large data sets can be transformed, generalized, simplified, and presented in different ways to influence how individuals interpret and understand the underlying information.
- **8.1.12.AP.1** Individuals evaluate and select algorithms based on performance, reusability, and ease of implementation.
- **8.1.12.AP.2** Programmers choose data structures to manage program complexity based on functionality, storage, and performance trade-offs.
- **8.1.12.AP.3** Trade-offs related to implementation, readability, and program performance are considered when selecting and combining control structures.
- **8.1.12.AP.4** Complex programs are designed as systems of interacting modules, each with a specific role, coordinating for a common overall purpose. Modules allow for better management of complex tasks.
- **8.1.12.AP.5** Complex programs are developed, tested, and analyzed by teams drawing on the members' diverse strengths using a variety of resources, libraries, and tools.

Career-Ready Practices

CRP1: Act as a responsible and contributing citizen and employee.

CRP2: Apply appropriate academic and technical skills.

CRP3: Attend to personal health and financial well-being. CRP4: Communicate clearly and effectively and with reason. CRP5: Consider the environmental, social and economic impacts of decisions. CRP6: Demonstrate creativity and innovation. CRP7: Employ valid and reliable research strategies. CRP8: Utilize critical thinking to make sense of problems and persevere in solving them. CRP9: Model integrity, ethical leadership and effective management. CRP10: Plan education and career paths aligned to personal goals. CRP11: Use technology to enhance productivity. CRP12: Work productively in teams while using cultural global competence.	
Differentiation/Technology Integration	Assessments
Differentiation Students will come up with their own ideas for a project, so each student will have their own goals set by themselves (with teacher approval) Grades will be based on how effectively they meet their goals Technology Students can choose their own programming language for their project. Some suggested languages are: ● Java ● Python ● JavaScript Students will utilize IDEs during the development of their program, some examples are: ● Eclipse ● IntelliJ ● IDLE	This Stage will be assessed based on the students plan before they move on to the next stage. Teacher feedback will be provided during the process.

Stage 2: Designing

Big Ideas: *Course Objectives/ Content Statement(s)*

Students will understand...

- The role that Designing plays in the Software Development Life Cycle
- How a program can be designed without actually coding yet
- How an effective design will lead to a well written program

Essential Questions

What provocative questions will foster inquiry, understanding, and transfer of learning?

- How do I turn my plan into a program?
- What abstractions do I need to build my program?
- How will my program be broken up into classes?
- What will go into each class?
- What is pseudocode?

Enduring Understandings

What will students understand about the big ideas?

Students will understand that:

- Pseudocode is a way to plan code without actually writing it. It can be written in a code format but much less formal than actual code
- Everything about programming is built on abstractions. Deciding how to break up a large task into smaller more manageable tasks is a key aspect in a well written program
- Giving different parts of the program different responsibilities is another key part of a well written program. Each Object needs to be responsible for appropriate tasks, and keeping track of appropriate variables and data. Ensuring that classes are written logically also makes adding to and updating the program easier.
- Starting with the user's side of the program means that it is the programmers job to decide how to break up tasks and make decisions about different Object's responsibilities.

Areas of Focus: Proficiencies (New Jersey Student Learning Standards)

Students will:

- Decide what classes their program needs
- Decide what methods and variables should go in each class
- Write comments for each method explaining the purpose, inputs, and outputs
- Write pseudocode for any algorithms in methods

Activities

During this stage of the SDLC students will come up with a plan for their program that will meet the requirements of their plan from the Analyze stage.

Students will create a design for their program using pseudocode. The plan should include detail about how the program will be broken up into classes, and what should go into each class.

NJ Computer Science and Design Thinking Standards

- **8.1.12.CS.3** Successful troubleshooting of complex problems involves multiple approaches including research, analysis, reflection, interaction with peers, and drawing on past experiences.
- **8.1.12.IC.1** The design and use of computing technologies and artifacts can positively or negatively affect equitable access to information and opportunities.
- **8.1.12.DA.1** Individuals select digital tools and design automated processes to collect, transform, generalize, simplify, and present large data sets in different ways to influence how other people interpret and understand the underlying information.
- **8.1.12.DA.2** Choices individuals make about how and where data is organized and stored affects cost, speed, reliability, accessibility, privacy, and integrity.
- **8.1.12.DA.3** Large data sets can be transformed, generalized, simplified, and presented in different ways to influence how individuals interpret and understand the underlying information.
- **8.1.12.AP.1** Individuals evaluate and select algorithms based on performance, reusability, and ease of implementation.
- **8.1.12.AP.2** Programmers choose data structures to manage program complexity based on functionality, storage, and performance trade-offs.
- **8.1.12.AP.3** Trade-offs related to implementation, readability, and program performance are considered when selecting and combining control structures.
- **8.1.12.AP.4** Complex programs are designed as systems of interacting modules, each with a specific role, coordinating for a common overall purpose. Modules allow for better management of complex tasks.

In this Stage students will review their plan with a classmate or the teacher to check for any logical problems before continuing.

Next students should write comments for the methods that go into their classes. The comments should include what each parameter is for, what the purpose of the method is, and what it returns.

For any methods that contain an algorithm specific to the program, students will write pseudocode to think through the algorithm.

● 8.1.12.AP.5 Complex programs are developed, tested, and analyzed by teams drawing on the members’ diverse strengths using a variety of resources, libraries, and tools. Career-Ready Practices CRP1: Act as a responsible and contributing citizen and employee. CRP2: Apply appropriate academic and technical skills. CRP3: Attend to personal health and financial well-being. CRP4: Communicate clearly and effectively and with reason. CRP5: Consider the environmental, social and economic impacts of decisions. CRP6: Demonstrate creativity and innovation. CRP7: Employ valid and reliable research strategies. CRP8: Utilize critical thinking to make sense of problems and persevere in solving them. CRP9: Model integrity, ethical leadership and effective management. CRP10: Plan education and career paths aligned to personal goals. CRP11: Use technology to enhance productivity. CRP12: Work productively in teams while using cultural global competence.	
Differentiation/Technology Integration	Assessments
Differentiation Students will come up with their own ideas for a project, so each student will have their own goals set by themselves (with teacher approval) Grades will be based on how effectively they meet their goals Technology Students can choose their own programming language for their project. Some suggested languages are: ● Java ● Python ● JavaScript	● Graded project design ● Teacher provided feedback and checkpoints during the design process

Students will utilize IDEs during the development of their program, some examples are:	
--	--

- Eclipse
- IntelliJ
- IDLE

Stage 3: Coding

Big Ideas: *Course Objectives/ Content Statement(s)*

Students will understand...

- The role that Coding plays in the Software Development Life Cycle
- Coding is essential to creating a program, but there are many other aspects

Essential Questions

What provocative questions will foster inquiry, understanding, and transfer of learning?

- How can I implement my design?
- What additional information do I need to write the code for my program?

Enduring Understandings

What will students understand about the big ideas?

Students will understand that:

- Implementing the design after thinking through algorithms logically can make programing easier
- No programmer knows every command or method. It is almost always necessary to use online resources to find out what is possible in each language
- Collaboration with another programmer is helpful to think through logical problems, and when looking for a new method to accomplish a task

Areas of Focus: Proficiencies (New Jersey Student Learning Standards)

Students will:

- Write code to implement their plan
- Research additional functions/methods/commands that are needed
- Collaborate with a partner when needed

NJ Computer Science and Design Thinking Standards

- **8.1.12.CS.3** Successful troubleshooting of complex problems involves multiple approaches including research, analysis, reflection, interaction with peers, and drawing on past experiences.
- **8.1.12.IC.1** The design and use of computing technologies and artifacts can positively or negatively affect equitable access to information and opportunities.

Activities

In this Stage students will write code to implement their plan. It is possible that students may need to go back to modify their design during this stage if they realize there is a better way to accomplish a task, or if they want to add a new piece to their program. Students should update their design first before coding if that happens.

While coding students should work with each other to work through logic and check code when errors come up.

Students should be careful to follow their plan, and update any comments for methods as needed. Code should be clear to anyone who reads it, leaving no questions about what a segment of code does.

- **8.1.12.DA.1** Individuals select digital tools and design automated processes to collect, transform, generalize, simplify, and present large data sets in different ways to influence how other people interpret and understand the underlying information.
- **8.1.12.DA.2** Choices individuals make about how and where data is organized and stored affects cost, speed, reliability, accessibility, privacy, and integrity.
- **8.1.12.DA.3** Large data sets can be transformed, generalized, simplified, and presented in different ways to influence how individuals interpret and understand the underlying information.
- **8.1.12.AP.1** Individuals evaluate and select algorithms based on performance, reusability, and ease of implementation.
- **8.1.12.AP.2** Programmers choose data structures to manage program complexity based on functionality, storage, and performance trade-offs.
- **8.1.12.AP.3** Trade-offs related to implementation, readability, and program performance are considered when selecting and combining control structures.
- **8.1.12.AP.4** Complex programs are designed as systems of interacting modules, each with a specific role, coordinating for a common overall purpose. Modules allow for better management of complex tasks.
- **8.1.12.AP.5** Complex programs are developed, tested, and analyzed by teams drawing on the members' diverse strengths using a variety of resources, libraries, and tools.

Career-Ready Practices

CRP1: Act as a responsible and contributing citizen and employee.

CRP2: Apply appropriate academic and technical skills.

CRP3: Attend to personal health and financial well-being.

CRP4: Communicate clearly and effectively and with

reason. CRP5: Consider the environmental, social and economic impacts of decisions. CRP6: Demonstrate creativity and innovation. CRP7: Employ valid and reliable research strategies. CRP8: Utilize critical thinking to make sense of problems and persevere in solving them. CRP9: Model integrity, ethical leadership and effective management. CRP10: Plan education and career paths aligned to personal goals. CRP11: Use technology to enhance productivity. CRP12: Work productively in teams while using cultural global competence.	
Differentiation/Technology Integration	Assessments
Differentiation Students will come up with their own ideas for a project, so each student will have their own goals set by themselves (with teacher approval) Grades will be based on how effectively they meet their goals Technology Students can choose their own programming language for their project. Some suggested languages are: ● Java ● Python ● JavaScript Students will utilize IDEs during the development of their program, some examples are: ● Eclipse ● IntelliJ ● IDLE	● Graded program ○ Students should be assessed on how well they implemented their plan, not necessarily if their program functions perfectly ● Teacher will provide feedback throughout the coding process

Stage 4: Testing and Debugging

Big Ideas: *Course Objectives/ Content Statement(s)*

Students will understand...

- The role that Testing and Debugging plays in the Software Development Life Cycle
- Testing and Debugging is a cycle the must be carefully executed for an effective program

Essential Questions

What provocative questions will foster inquiry, understanding, and transfer of learning?

- How can I improve my existing code?
- How can I ensure that a user will not find problems in my program?

Enduring Understandings

What will students understand about the big ideas?

Students will understand that:

- Debugging and Testing can lead to starting the Design stage again for some parts of the program
- Testing is the process of running the program, ensuring the each line of code has run at some point
- Debugging is the process of fixing errors in code after testing
- There are three types of errors in coding
 - Syntax errors are when a word is spelled incorrectly, parameters are used improperly, or variables are given invalid values. These errors will be caught during compiling, or with an IDE as the programmer is coding.
 - Runtime errors occur when the syntax is good, but something happens while running the program that causes a mathematical error, a value to have an invalid value, memory to run out, or an infinite loop or recursion.
 - Logical errors occur when the program functions, but the output is not what was intended

Areas of Focus: Proficiencies (New Jersey Student Learning Standards)

Students will:

- Run their programs and look for errors
- Test several different cases to ensure each line of the program has been tested

Activities

In this Stage students will run and test their programs to look for any problems the user may experience after the program is complete. It is natural for the programmer to need to go back to the Design stage for part of the program if an error occurs.

- Attempt to cause problems in their programs to see what happens if the user makes a mistake
- Collaborate with a partner when needed

NJ Computer Science and Design Thinking Standards

- **8.1.12.CS.3** Successful troubleshooting of complex problems involves multiple approaches including research, analysis, reflection, interaction with peers, and drawing on past experiences.
- **8.1.12.IC.1** The design and use of computing technologies and artifacts can positively or negatively affect equitable access to information and opportunities.
- **8.1.12.DA.1** Individuals select digital tools and design automated processes to collect, transform, generalize, simplify, and present large data sets in different ways to influence how other people interpret and understand the underlying information.
- **8.1.12.DA.2** Choices individuals make about how and where data is organized and stored affects cost, speed, reliability, accessibility, privacy, and integrity.
- **8.1.12.DA.3** Large data sets can be transformed, generalized, simplified, and presented in different ways to influence how individuals interpret and understand the underlying information.
- **8.1.12.AP.1** Individuals evaluate and select algorithms based on performance, reusability, and ease of implementation.
- **8.1.12.AP.2** Programmers choose data structures to manage program complexity based on functionality, storage, and performance trade-offs.
- **8.1.12.AP.3** Trade-offs related to implementation, readability, and program performance are considered when selecting and combining control structures.
- **8.1.12.AP.4** Complex programs are designed as systems of interacting modules, each with a specific role, coordinating for a common overall

Students will come up with several different test cases to make sure each line of their program has been executed. If necessary each line of the program may need to be executed multiple times to make sure different types of values are tested for different variables.

Students should keep a log of whenever they need to go back to the Design stage for a part of their program. It is important to make sure the program is written as effectively as possible, even if that means going back to the beginning for some parts of the program. The goal is to make the finished product as efficient and logical as possible, not to minimize programming time.

purpose. Modules allow for better management of complex tasks. ● 8.1.12.AP.5 Complex programs are developed, tested, and analyzed by teams drawing on the members’ diverse strengths using a variety of resources, libraries, and tools. Career-Ready Practices CRP1: Act as a responsible and contributing citizen and employee. CRP2: Apply appropriate academic and technical skills. CRP3: Attend to personal health and financial well-being. CRP4: Communicate clearly and effectively and with reason. CRP5: Consider the environmental, social and economic impacts of decisions. CRP6: Demonstrate creativity and innovation. CRP7: Employ valid and reliable research strategies. CRP8: Utilize critical thinking to make sense of problems and persevere in solving them. CRP9: Model integrity, ethical leadership and effective management. CRP10: Plan education and career paths aligned to personal goals. CRP11: Use technology to enhance productivity. CRP12: Work productively in teams while using cultural global competence.	
Differentiation/Technology Integration	Assessments
Differentiation Students will come up with their own ideas for a project, so each student will have their own goals set by themselves (with teacher approval) Grades will be based on how effectively they meet their goals Technology Students can choose their own programming language for their project. Some suggested languages are: ● Java ● Python	● Graded program ● Graded debugging log ● Teacher will provide feedback throughout the testing and debugging process

- JavaScript

Students will utilize IDEs during the development of their program, some examples are:

- Eclipse
- IntelliJ
- IDLE

Stage 5: Maintaining	
Big Ideas: <i>Course Objectives/ Content Statement(s)</i> Students will understand... • The role that Maintaining plays in the Software Development Life Cycle • Maintaining contains the previous 4 stages of the Software Development Life Cycle	
Essential Questions <i>What provocative questions will foster inquiry, understanding, and transfer of learning?</i>	Enduring Understandings <i>What will students understand about the big ideas?</i>
• How can my program be improved? • What improvements should take priority over others?	Students will understand that: • Maintaining is the longest part of a program's life • Programs constantly need to be updated to meet user's needs, compete with other programs, or fix bugs that may not have been caught in testing
Areas of Focus: Proficiencies (New Jersey Student Learning Standards)	Activities
Students will: • Provide feedback to at least one classmate to suggest an update to their program • Pick at least one suggestion to create an update to their program, creating a new Analyze stage • Create a design for the update and implement it NJ Computer Science and Design Thinking Standards • 8.1.12.CS.3 Successful troubleshooting of complex problems involves multiple approaches including research, analysis, reflection, interaction with peers, and drawing on past experiences. • 8.1.12.IC.1 The design and use of computing technologies and artifacts can positively or negatively affect equitable access to information and opportunities. • 8.1.12.DA.1 Individuals select digital tools and design automated processes to collect, transform, generalize, simplify, and present large data sets in different ways to influence how other people	In this Stage students will give feedback to other students to suggest an update to their program. This could be fixing a bug, or adding new functionality, or both. Out of the suggestions a student receives, they will create a new plan for an update to their program starting with the Analyze stage. Then students will create a new Design like they did at the start of their project. Students will then move on to the Coding, then Testing and Debugging stages in the same way they did when they started creating their program.

interpret and understand the underlying information.

- **8.1.12.DA.2** Choices individuals make about how and where data is organized and stored affects cost, speed, reliability, accessibility, privacy, and integrity.
- **8.1.12.DA.3** Large data sets can be transformed, generalized, simplified, and presented in different ways to influence how individuals interpret and understand the underlying information.
- **8.1.12.AP.1** Individuals evaluate and select algorithms based on performance, reusability, and ease of implementation.
- **8.1.12.AP.2** Programmers choose data structures to manage program complexity based on functionality, storage, and performance trade-offs.
- **8.1.12.AP.3** Trade-offs related to implementation, readability, and program performance are considered when selecting and combining control structures.
- **8.1.12.AP.4** Complex programs are designed as systems of interacting modules, each with a specific role, coordinating for a common overall purpose. Modules allow for better management of complex tasks.
- **8.1.12.AP.5** Complex programs are developed, tested, and analyzed by teams drawing on the members' diverse strengths using a variety of resources, libraries, and tools.

Career-Ready Practices

CRP1: Act as a responsible and contributing citizen and employee.

CRP2: Apply appropriate academic and technical skills.

CRP3: Attend to personal health and financial well-being.

CRP4: Communicate clearly and effectively and with reason.

CRP5: Consider the environmental, social and economic impacts of decisions.

CRP6: Demonstrate creativity and innovation.

CRP7: Employ valid and reliable research strategies. CRP8: Utilize critical thinking to make sense of problems and persevere in solving them. CRP9: Model integrity, ethical leadership and effective management. CRP10: Plan education and career paths aligned to personal goals. CRP11: Use technology to enhance productivity. CRP12: Work productively in teams while using cultural global competence.	
Differentiation/Technology Integration	Assessments
Differentiation Students will come up with their own ideas for a project, so each student will have their own goals set by themselves (with teacher approval) Grades will be based on how effectively they meet their goals Technology Students can choose their own programming language for their project. Some suggested languages are: ● Java ● Python ● JavaScript Students will utilize IDEs during the development of their program, some examples are: ● Eclipse ● IntelliJ ● IDLE	● Graded updates to program ● Suggestions to improve other students' programs ● Design plan for updates ● Teacher will provide feedback throughout the maintaining process