|  |
| --- |
| **Semester 1** |
| **Unit 1: Digital Information** | **A/B Schedule of Classes Duration: 3 weeks**  |
| **Lesson Progression** | **Key Questions** | **Instructional Strategies** | **Assessment** |
| L1: Welcome to CSPL2: Representing InformationL3: Circle Square PatternsL4: Binary NumbersL5: Overflow and RoutingL6: Representing TextL7: Black and White Images L8: Color ImagesL9: Lossless CompressionL10: Lossy CompressionL11: Intellectual PropertyL12: Project Digital Information DilemmasL13: Project Digital Information DilemmasL14: Unit Assessment | * Are the ways in which digital information is encoded more laws of nature or man-made?
* What kinds of limitations does the binary encoding of information impose on what can be represented inside a computer?
* How accurately can human experience and perception be captured or reflected in digital information?
 | * Journaling
* Peer Feedback
* Classroom Discussions
* Think-Pair-Share
* Pair Programming
* Debugging
* Unplugged/Plugged Activities
 | **Formal:*** PA = Performance Assessment from Code.org online widget
* SA = Summative Chapter/Unit Assessment
* O: Observation

**Informal:*** HW: Hands-on work from scholars
* IO: Interactive and non-interactive Observation

**Resources:*** Code.org online resources
 |
| **APCSP Standards** |
| 2-DA-07 2-IC-20 3A-AP-213A-CS-02 3A-DA-09 3A-DA-10 3A-IC-28 3A-IC-24 3B-IC-27  | - Represent data using multiple encoding schemes.- Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options.- Evaluate and refine computational artifacts to make them more usable and accessible.- Compare levels of abstraction and interactions between application software, system software and hardware layers.- Translate between different bit representations of real-world phenomena, such as characters, numbers, and images.- Evaluate the tradeoffs in how data elements are organized and where data is stored.- Explain the beneficial and harmful effects that intellectual property laws can have on innovation.- Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices- Predict how computational innovations that have revolutionized aspects of our culture might evolve. |
| **Computational Thinking Practices** |
| * P1 Connecting Computing
* P4 Analyzing Problems & Artifacts
 |

|  |
| --- |
| **Semester 1** |
| **Unit 2: Digital Information** | **A/B Schedule of Clases Duration: 2 weeks**  |
| **Lesson Progression** | **Key Questions** | **Instructional Strategies** | **Assessment** |
| L1: Welcome to he InternetL2: Building a NetworkL3: Need for AddressingL4: Routers and RedundancyL5: PacketsL6: HTTP and DNSL7: Project Internet DilemmasL8: Project Internet DilemmasL9 Unit Assessment |  | * Journaling
* Peer Feedback
* Classroom Discussions
* Think-Pair-Share
* Pair Programming
* Debugging
* Unplugged/Plugged Activities
 | **Formal:*** PA = Performance Assessment from Code.org online widget
* SA = Summative Chapter/Unit Assessment
* O: Observation

**Informal:*** HW: Hands-on work from scholars
* IO: Interactive and non-interactive Observation

**Resources:*** Code.org online resources
 |
| **APCSP Standards** |
| 2-NI-04 3A-IC-24 3A-IC-28 3A-IC-30 3B-IC-26 3B-IC-28 3B-NI-03  | - Model the role of protocols in transmitting data across networks and the Internet.- Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.- Explain the beneficial and harmful effects that intellectual property laws can have on innovation.- Evaluate the social and economic implications of privacy in the context of safety, law, or ethics.- Evaluate the impact of equity, access, and influence on the distribution of computing resources in a global society.- Debate laws and regulations that impact the development and use of software 3A-NI-04 - Evaluate the scalability and reliability of networks, by describing the relationship between routers, switches, servers, topology, and addressing.- Describe the issues that impact network functionality (e.g., bandwidth, load, delay, topology).. |
| **Computational Thinking Practices** |
| * P2 Creating Computational Artifacts
* P3 Abstracting
 |

|  |
| --- |
| **Semester 1** |
| **Unit 3: Intro to APP Design** | **A/B Schedule for Classes Duration: 3 weeks**  |
| **Lesson Progression** | **Key Questions** | **Instructional Strategies** | **Assessment** |
| L1: Introduction to APPsL2: Introduction to Design ModeL3: Project Designing an APP Part 1L4: Designing an APP Part 2L5: The Need for Programming LanguagesL6: Intro to ProgrammingL7: DebuggingL8: Project Designing an App Part 3L9: Project Designing an APP Part 4L10 Project Designing an APP Part 5L11: Unit Assessment |  | * Journaling
* Peer Feedback
* Classroom Discussions
* Think-Pair-Share
* Pair Programming
* Debugging
* Unplugged/Plugged Activities
 | **Formal:*** PA = Performance Assessment from Code.org online widget
* SA = Summative Chapter/Unit Assessment
* O: Observation

**Informal:*** HW: Hands-on work from scholars
* IA: Interactive and non-interactive Observation

**Resources:*** Code.org online resources
 |
| **CTSA Standards** |
| 3A-AP-133A-AP-15 3A-AP-16 3A-AP-17 3A-AP-19 3A-AP-21 3A-AP-22 3A-AP-23 3B-AP-14 3A-CS-03 | - Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.- Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance and explain the benefits and drawbacks of choices made.- Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.- Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.- Systematically design and develop programs for broad audiences by incorporating feedback from users.- Evaluate and refine computational artifacts to make them more usable and accessible.- Design and develop computational artifacts working in team roles using collaborative tools.- Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs. - Construct solutions to problems using student-created components, such as procedures, modules and/or objects.- Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors. |
| **Computational Thinking Practices** |
| * P1 Connecting Computing
* P3 Abstracting
* P4 Analyzing Problems & Artifacts
 |

|  |
| --- |
| **Semester 1** |
| **Unit 4: Variables, Conditions and Functions** | **A/B Schedule for Classes Duration: 6 weeks**  |
| **Lesson Progression** | **Key Questions** | **Instructional Strategies** | **Assessment** |
| L1: Variables ExploreL2: Variable InvestigateL3: Variables PracticeL4: Variables MakeL5: Conditionals ExploreL6: Conditionals InvestigateL7: Conditionals PracticeL8: Conditionals MakeL9: Functions Explore / InvestigateL10: Functions PracticeL11: Functions MakeL12: Project Decision Maker APP Part 1L13: Project Decision Maker APP Part 2L14: Project Decision Maker APP Part 3L15: Unit Assessment |  | * Journaling
* Peer Feedback
* Classroom Discussions
* Think-Pair-Share
* Pair Programming
* Debugging
* Unplugged/Plugged Activities
 | **Formal:*** PA = Performance Assessment from Code.org online widget
* SA = Summative Chapter/Unit Assessment
* O: Observation

**Informal:*** HW: Hands-on work from scholars
* IO: Interactive and non-interactive Observation

**Resources:*** Code.org online resources
 |
| **APCSP Standards** |
| 2-AP-10 2-AP-11 2-AP-12 2-AP-19 3A-AP-15 3A-AP-16 3A-AP-17 3B-AP-14 3B-AP-21 3B-AP-23  | - Use flowcharts and/or pseudocode to address complex problems as algorithms.- Create clearly named variables that represent different data types and perform operations on their values.- Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.- Document programs in order to make them easier to follow, test, and debug.- Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance and explain the benefits and drawbacks of choices made.- Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.- Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.- Construct solutions to problems using student-created components, such as procedures, modules and/or objects.- Develop and use a series of test cases to verify that a program performs according to its design specifications.- Evaluate key qualities of a program through a process such as a code review |
| **Computational Thinking Practices** |
| * P1 Connecting Computing
* P2 Creating Computational Artifacts
* P4 Analyzing Problems & Artifacts
 |

**END OF SEMESTER 1**

|  |
| --- |
| **Semester 2** |
| **Unit 5: Lists, Loops and Traversals** | **A/B Schedule for Classes Duration: 4 weeks** |
| **Lesson Progression** | **Key Questions** | **Instructional Strategies** | **Assessment** |
| L1: Lists ExploreL2: Lists InvestigateL3: Lists PracticeL4: Lists MakeL5: Loops ExploreL6: Loops InvestigateL7: Loops PracticeL8: Loops MakeL9: Traversals ExploreL10: Traversals InvestigateL11: Traversals PracticeL12: Traversals MakeL13: Project Hackathon Part 1L14: Project Hackathon Part 2L15: Project Hackathon Part 3L16: Project Hackathon Part 4L17: Project hackathon Part 5L18: Unit Assessment |  | * Journaling
* Peer Feedback
* Classroom Discussions
* Think-Pair-Share
* Pair Programming
* Debugging
* Unplugged/Plugged Activities
 | **Formal:*** PA = Performance Assessment from Code.org online widget
* SA = Summative Chapter/Unit Assessment
* O: Observation

**Informal:*** HW: Hands-on work from scholars
* IA: Interactive and non-interactive observation

**Resources:*** Code.org online resources
 |
| **APCSP Standards** |
|  |  |
| **Computational Thinking Practices** |
| * P1 Connecting Computing
* P2 Creating Computational Artifacts
* P3 Abstracting
* P4 Analyzing Problems and Artifacts
 |

|  |
| --- |
| **Semester 2** |
| **Unit 6: Algorithms** | **A/B Schedule for Classes Duration: 2 weeks** |
| **Lesson Progression** | **Key Questions** | **Key Concepts and Pedagogy** | **Assessment** |
| L1: Algorithms Solve ProblemsL2: Algorithm EfficiencyL3: Unreasonable TimeL4: Limits of AlgorithmsL5: Parallel and Distributed AlgorithmsL6: Unit Assessment |  | * Journaling
* Peer Feedback
* Classroom Discussions
* Think-Pair-Share
* Pair Programming
* Debugging
* Unplugged/Plugged Activities
 | **Formal:*** PA = Performance Assessment from Code.org online widget
* SA = Summative Chapter/Unit Assessment
* O: Observation

**Informal:*** O: Hands-on work from scholars
* O: Interactive and non-interactive observation

**Resources:*** Code.org online resources
 |
| **APCSP Standards** |
|  |  |
| **Computational Thinking Practices** |
| * P1 Connecting Computing
* P2 Creating Computational Artifacts
* P3 Abstracting
* P4 Analyzing Problems and Artifacts
 |

|  |
| --- |
| **Semester 2** |
| **Unit 7: Parameters, Returns and Libraries** | **A/B Schedule for Classes Duration: 3 weeks** |
| **Lesson Progression** | **Key Questions** | **Key Concepts and Pedagogy** | **Assessment** |
| L1: Parameters and Returns ExploreL2: Parameters and Return InvestigateL3: Parameters and Return PracticeL4: Parameters and Return MakeL5: Libraries ExploreL6: Libraries InvestigateL7: Libraries PracticeL8: Project Make a Library Part 1L9: Project Make a Library Part 2L10: Project Make a Library Part 3L11: Unit Assessment |  | * Journaling
* Peer Feedback
* Classroom Discussions
* Think-Pair-Share
* Pair Programming
* Debugging
* Unplugged/Plugged Activities
 | **Formal:*** PA = Performance Assessment from Code.org online widget
* SA = Summative Chapter/Unit Assessment
* O: Observation

**Informal:*** O: Hands-on work from scholars
* O: Interactive and non-interactive observation

**Resources:*** Code.org online resources
 |
| **APCSP Standards** |
|  |  |
| **Computational Thinking Practices** |
| * P1 Connecting Computing
* P2 Creating Computational Artifacts
* P3 Abstracting
* P4 Analyzing Problems and Artifacts
 |

|  |
| --- |
| **Semester 2** |
| **Unit 8: Create PT Prep** | **A/B Schedule for Classes Duration: 2 weeks** |
| **Lesson Progression** | **Key Questions** | **Key Concepts and Pedagogy** | **Assessment** |
|  |  | * Journaling
* Peer Feedback
* Classroom Discussions
* Think-Pair-Share
* Pair Programming
* Debugging
* Unplugged/Plugged Activities
 | **Formal:*** PA = Performance Assessment from Code.org online widget
* SA = Summative Chapter/Unit Assessment
* O: Observation

**Informal:*** O: Hands-on work from scholars
* O: Interactive and non-interactive observation

**Resources:*** Code.org online resources
 |
| **APCSP Standards** |
|  |  |
| **Computational Thinking Practices** |
| * P1 Connecting Computing
* P2 Creating Computational Artifacts
* P3 Abstracting
* P4 Analyzing Problems and Artifacts
 |