COMPUTATIONAL THINKING AND PROBLEM SOLVING

COURSE DESCRIPTION:

Computational Thinking and Problem Solving (CTPS) is a course unlike many other traditional lecture-based classes. The course is designed to be a yearlong class for high school students (ideally 10th grade students who have digital literacy skills). The curriculum consists of six team-based projects and invite the students the opportunity to gain experience and develop proficiency with complex problem-solving skills and techniques.

Computational Thinking is a methodology for solving complex problems. The problem statement is formulated and its solution expressed in such a way that a computer—human or machine—can effectively carry it out.[1] Its utility lies in the ability to algorithmically solve complicated problems of scale; problems which could not be solved otherwise, and/or problems for which alternative strategies require extraordinary amounts of time and resources. Computational Thinking is rooted in a three-stage iterative process (1) problem formulation or abstraction, (2) solution expression or automation, and (3) solution execution, evaluation and analysis.

The history of computational thinking dates back at least to the 1950s but many of its component concepts are much older.[2] The term *computational thinking* was first used by Seymour Papert in 1980[3] and again in 1996.[3] In 2006, a popular paper, published by Jeannette Wing, categorized computational thinking as a critical skill and proposed that it become an essential part of every child's education.[4] Valerie Barr and Chris Stephenson published a paper in 2011 suggesting that computational thinking be a fundamental skill integrated across disciplines into a variety of STEM courses.[5] Conrad Wolfram argued, strongly, computational thinking was important enough that it should be a course/subject unto itself.[6]

This course presents computational thinking in the framework of a team-based, workplace oriented project based learning course. The students experience a unique sense of empowerment, operating as independent project teams (guided and mentored) by the classroom teacher. The students many of them for the first time, practice and gain proficiency in problem solving, critical thinking, reflection and metacognition, writing and presentation skills, and project teams, and peer review. The course seeks to engage students and give students a sense of ownership and a maker mentality. The exercises require basic digital literacy (Microsoft Word, Excel, PowerPoint) and benefit a broad range of students with widely varying knowledge, experience, skills, and abilities.

STUDENT LEARNING OUTCOMES

Computational Thinking

- Using abstractions and pattern recognition to represent problems in new and different ways
- Logically organizing and analyzing data
- Breaking down complex problems down into smaller solvable parts
- Approaching problems using programmatic thinking techniques such as iteration, symbolic representation, and logical operations
- Reformulating problems into a series of ordered steps (algorithmic thinking)
- Identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources
- Generalizing problem-solving process adapting existing knowledge to assist in solving new problems.

Problem Solving

- Clarifying the definition of a problem by identifying the specifics goals of an acceptable solution
- Persist in working with difficult problems
- Tolerate and manage ambiguity
- Describing and generating data which enables a clearer understanding of a problem
- Formulating challenge questions which invite solutions and encourage brainstorming
- Generating ideas that address the aforementioned challenge questions
- Formulating solutions and analyzing the appropriateness and quality of the answer

Digital / Computer Literacy

- Create, format, and print professional quality documents.
- Create and incorporate a cover page, table of contents, headers, footers, and footnotes into digital documents.
- Create and insert charts and graphs into digital documents.
- Create, sort, filter and manipulate professional quality pivot tables.
- Create, print and present professional quality presentations.
- Create enhanced quality presentations feel using publicly available templates.
- Modify existing algorithms created using block programming utilities.
- Create and implement new algorithms created using block programming utilities.
- Modify existing algorithms created using formal programming/scripting languages.
- Modify existing web page content (adding / changing /deleting text, images, videos).
- Familiarity with the characteristics and behaviors associated with clean computing
- Knowledge of best practices with regard to computer use and digital security.
- Awareness of key issues and current topics in social networking tools and communities.

Professional Skills

- Ability to work independently with minimal guidance and direction.
- Ability to recognize when, where and how to ask for assistance.
- Ability to work collaboratively on a common assignment.
- Ability to work with peers on a group project.
- Ability to work constructively with subordinates on a group project.
- Ability to communicate both orally, and writing according to professional standards.
- Ability to work in a variety of project teams of different configuration.

INSTRUCTOR

The instructor for this course is:

- Email:
- Phone:

ATTENDANCE:

The team-based structure of this course requires your attendance, **without exception**. You will need to be present at every class, prepared and ready to work. In other words, readings or other assignments need to be completed before the start of a discussion. You, as an individual, are responsible for all of the material presented in class.

If your absence from class is unavoidable, your instructor needs to be notified **before class starts.** The class instructor has **full and final authority** to decide **whether, and on what terms,** a student is permitted to make up work missed through absence.

Class attendance does not, by itself, guarantee a passing grade. Your team's project deliverable, as well as project team's assessment of your contribution, will determine your grade.

STUDENT RESPONSIBILITY AND REQUIREMENTS:

You are expected to take an active role in your learning. It is recommended that you take notes on every reading/video and study them afterwards. At a minimum, you will need to:

- Read/watch all assigned material on the date it is scheduled
- Take notes on all assigned materials
- Complete the projects as assigned
- Work in teams to solve problems
- Complete the project deliverables and submit them as assigned
- Carry out the required steps for writing assignments

METHODS OF INSTRUCTION:

Computational Thinking and Problem Solving (CTPS) is a course unlike many other traditional lecture-based classes. Students are organized into teams, and are required to be active learners who take ownership and in control of their learning. The class exercises leverage, and depend on, a certain level of student preparedness before class. Class time is an opportunity for the teams to translate knowledge and theory into practice under the guidance and mentorship of the teacher.

CTPS is rooted in Problem Based Learning (PBL) as the central pedagogy for the course. Students are assembled into project teams (6 different projects, 6 different teams) over the course of the year. The projects/problems are modeled after authentic workplace-based exercises. The teams are asked to take an inventory of “what they already know”, clearly define “what they need to discover”, and document a plan for “accessing new information” in order to “complete the project and solve the problem”. Students “learn how to analyze, criticize and select from alternative sources of information and courses of action; how to think about problems that may have more than one viable solution; how to work together with those of differing views; and how to confront and act upon problems and situations in constructive and creative ways.”

The teacher, acting as a facilitator, provides guidance and mentorship but not instruction. They facilitate learning by providing appropriate scaffolding of that process and measures student success via outcome based assessments. This methodology is highly transferable and builds the cognitive ability to apply knowledge gained to any situation or problem statement. Students have an opportunity to learn via application and understanding, rather than testing and memorization.

LABORATORY AND HOMEWORK:

Your assignments will need to be professional in their format and presentation, and completed thoroughly in accordance with the instructions. Assignments should be completed properly and thoroughly and be ready to send in on or before the assigned due date. **NO LATE ASSIGNMENTS ARE ACCEPTED!**

All of the assignments will be explained in class and you will have the opportunity to ask questions at that time.

INSTRUCTOR'S NOTE:

The instructor reserves the right to make changes to this syllabus at any time during the semester. A new syllabus may or may not be distributed at the discretion of the instructor.

EXERCISES:

A letter grade will be awarded at the completion of the course according to the following weighted average:

Project 1: System Design and Proposal Development (20%)

Students work in project teams and are responsible for the design of the digital infrastructure for the company's gaming lounge. Each project team will develop and present a proposal (design and budget) for this important new addition to the corporate headquarters.

Computational Thinking Assessment (STAIR Chart and Rubric)	100
Excel Workbook	20
Word Report	20
Networking Assignment	20
Other Assignments and Writing Reflection	20
Quality and Thoroughness of Presentation	20

Project 2: Data Analytics (20%)

Students work in project teams and are responsible for the development of an energy strategy for a home, or business. Each project team will develop and present a model of existing costs, and a plan for reducing those costs by 15%.

Computational Thinking Assessment (STAIR Chart and Rubric)	100
Excel Workbook	20
Pivot Table	20
Word Report	20
Other Assignments and Writing Reflection	20
Quality and Thoroughness of Presentation	20

Project 3: Website Design and Development (20%)

Students work in project teams and are responsible for the development of the organization's website. Each project team will design, develop and present a new website in accordance with the project's specifications

Computational Thinking Assessment (STAIR Chart and Rubric)	100
Non-Profit Planning Worksheet	20
Non-Profit Mockup and Wireframe	20
Non-Profit Website	20
Other Assignments and Writing Reflection	20
Quality and Thoroughness of Presentation	20

Project 4: Mobile Application Development (20%)

Students are assigned to a project team responsible for development of a mobile application. Each project team will create a presentation describing their application, a storyboard describing the logic of their design, and the application itself.

Computational Thinking Assessment (STAIR Chart and Rubric)	100
Presentation	20
Storyboard	20
Mobile Application	20
Other Assignments and Writing Reflection	20
Quality and Thoroughness of Presentation	20

Project 5: Privacy and Security (20%)

Students are assigned to a project team responsible for development of a multimedia presentation designed to teach others the fundamentals of CyberSecurity.

Computational Thinking Assessment (STAIR Chart and Rubric)	100
Multimedia Presentation	60
Other Assignments and Writing Reflection	20
Quality and Thoroughness of Presentation	20
TOTAL:	1,000

GRADING:

Student performance will be assessed both objectively and subjectively. Each team will receive a grade for project deliverables and presentations. Individual grades will be assessed for each student, using their team's grade as a baseline, with adjustments for individual attendance, self-preparedness and personal contribution to the team deliverables.

The point to Letter Grade equivalency is as follows:

Letter Grade	Project Score	Cumulative Score
A	94 – 100	940 – 1000
A-	90 – 93	900 – 939
B+	87 – 89	870 – 899
B	83 – 86	830 – 869
B-	80 – 82	800 – 829
C+	77 – 79	770 – 799
C	70 – 76	700 – 769
D	60 – 69	600 – 699
F	< 60	< 600

Students, who do not fulfill all of the requirements of the course, will be given an Incomplete.

COMPLIANCE with Massachusetts Digital Literacy and Computer Science Framework

Computing and Society		
9-12.CAS.a.2	Explain safe practices when collaborating online, including how to anticipate potentially dangerous situations.	5: Lesson 22 - 27
9-12.CAS.a.3	Construct strategies to combat cyberbullying/harassment.	5: Lesson 11, Lesson 22 -27
9-12.CAS.a.4	Identify the mental health consequences of cyberbullying/harassment.	5: Lesson 5, Lesson 11
9-12.CAS.a.5	Explain how peer pressure in social computing settings influences choices.	5: Lesson 5, Lesson 11
9-12.CAS.a.6	Apply strategies for managing negative peer pressure and encouraging positive peer pressure.	5: Lesson 11, Lesson 22 - 27
9-12.CAS.b.1	Model mastery of the school's Acceptable Use Policy (AUP).	5: Lesson 3
9-12.CAS.b.2	Identify computer-related laws and analyze their impact on digital privacy, security, intellectual property, network access, contracts, and consequences of sexting and harassment.	3: Lesson 6 5: Lesson 2, Lesson 3, Lesson 5
9-12.CAS.b.3	Discuss the legal and ethical implications associated with malicious hacking and software piracy.	5: Lesson 19, Lesson 21
9-12.CAS.b.4	Interpret software license agreements and application permissions.	5: Lesson 3
9-12.CAS.c.1	Explain the impact of the digital divide on access to critical information.	2: Lesson 19
9-12.CAS.c.2	Discuss the impact of computing technology on business and commerce (e.g., automated tracking of goods, automated financial transaction, e-commerce, cloud computing).	2: Lesson 2, Lesson 4 4: Lesson 1 5: Lesson 6, Lesson 9
9-12.CAS.c.4	Create a digital artifact that is designed to be accessible (e.g., closed captioning for audio, alternative text for images).	3: Lesson 7, Lesson 21
9-12.CAS.c.5	Analyze the beneficial and harmful effects of computing innovations (e.g., social networking, delivery of news and other public media, intercultural communication).	2: Lesson 2, Lesson 5 5: Lesson 4, Lesson 5, Lesson 6, Lesson 16
9-12.CAS.c.6	Cultivate a positive web presence (e.g., digital resume, portfolio, social media).	5: Lesson 22 -27
9-12.CAS.c.7	Identify ways to use technology to support lifelong learning.	2: Lesson 19 3: Lesson 3 5: Lesson 16
9-12.CAS.c.8	Analyze the impact of values and points of view that are presented in media messages (e.g., racial, gender, political).	3: Lesson 4 5: Lesson 16
9-12.CAS.c.9	Discuss the social and economic implications associated with malicious hacking, software piracy, and cyber terrorism.	5: Lesson 19, Lesson 21

Digital Tools and Collaboration		
9-12.DTC.a.1	Use digital tools to design and develop a significant digital artifact (e.g., multipage website, online portfolio, simulation).	1: Lesson 14, Lesson 20, Lesson 29 2: Lesson 6, Lesson 11, Lesson 13, Lessons 22-24 Lesson 26 3: Lesson 32 4: Lessons 29-30
9-12.DTC.a.2	Select digital tools or resources based on their efficiency and effectiveness to use for a project or assignment and justify the selection.	1: Lesson 10 Lesson 13 3: Lesson 21
9-12.DTC.b.1	Communicate and publish key ideas and details to a variety of audiences using digital tools and media-rich resources.	1: Lessons 29-30 2: Lesson 13 Lessons 15-17 Lessons 23-24 Lesson 27 3: Lesson 24 Lessons 32-33 4: Lessons 29-30
9-12.DTC.b.2	Collaborate on a substantial project with outside experts or others through online digital tools (e.g., science fair project, community service project, capstone project).	3: Lesson 17, Lesson 21 Lesson 24 4: Lesson 9 Lesson 11
9-12.DTC.c.1	Generate, evaluate, and prioritize questions that can be researched through digital resources or tools.	1: Lesson 1 Lesson 2 Lesson 7 Lesson 9 2: Lesson 1 Lesson 4 3: Lessons 1-3
9-12.DTC.c.2	Perform advanced searches to locate information and/or design a data-collection approach to gather original data (e.g., qualitative interviews, surveys, prototypes, simulations).	2: Lessons 15-16 3: Lesson 2 Lesson 5
9-12.DTC.c.3	Evaluate digital sources needed to solve a given problem (e.g., reliability, point of view, relevancy).	1: Lesson 7 Lesson 9 3: Lesson 5
9-12.DTC.c.4	Gather, organize, analyze, and synthesize information using a variety of digital tools.	1: Lesson 7 Lesson 9 2: Lessons 5 – 8 Lesson 11 Lessons 21 – 24 3: Lesson 3 Lesson 7 Lesson 21
9-12.DTC.c.5	Create an artifact that answers a research question, communicates results and conclusions, and cites sources.	2: Lessons 21 - 24

Computing Systems		
9-12.CS.a.1	Select computing devices (e.g., probe, sensor, tablet) to accomplish a real-world task (e.g., collecting data in a field experiment) and justify the selection.	1: Lesson 10 Lesson 20
9-12.CS.a.2	Examine how the components of computing devices are controlled by and react to programmed commands.	4: Lesson 17
9-12.CS.a.5	Describe how computing devices manage and allocate shared resources [e.g., memory, Central Processing Unit (CPU)].	1: Lesson 6
9-12.CS.a.6	Examine the historical rate of change in computing devices (e.g., power/energy, computation capacity, speed, size, ease of use) and discuss the implications for the future.	5: Lesson 17
9-12.CS.b.1	Identify a problem that cannot be solved by humans or machines alone and design a solution for it by decomposing the task into sub-problems suited for a human or machine to accomplish (e.g., a human-computer team playing chess, forecasting weather, piloting airplanes).	2: Lesson 1 3: Lesson 17 Lesson 21 Lesson 22 Lesson 23 Lesson 25
9-12.CS.c.1	Explain how network topologies and protocols enable users, devices, and systems to communicate with each other.	1: Lesson 23
9-12.CS.c.2	Examine common network vulnerabilities (e.g., cyberattacks, identity theft, privacy) and their associated responses.	5: Lesson 14 Lesson 19 Lesson 21

Computational Thinking		
9-12.CT.a.1	Discuss and give an example of the value of generalizing and decomposing aspects of a problem in order to solve it more effectively.	0: Lesson 4 1: Lesson 3 Lesson 19 4: Lesson 1
9-12.CT.b.1	Recognize that the design of an algorithm is distinct from its expression in a programming language.	0: Lesson 4 1: Lesson 24 5: Lesson 17
9-12.CT.b.2	Represent algorithms using structured language, such as pseudocode.	0: Lesson 4 1: Lesson 24 5: Lesson 17
9-12.CT.b.3	Explain how a recursive solution to a problem repeatedly applies the same solution to smaller instances of the problem.	1: Lesson 13
9-12.CT.b.5	Explain that there are some problems, which cannot be computationally solved.	0: Lesson 5
9-12.CT.c.2	Create an appropriate multidimensional data structure that can be filtered, sorted, and searched (e.g., array, list, record).	5: Lesson 19 Lesson 22-29
9-12.CT.c.3	Create, evaluate, and revise data visualization for communication and knowledge.	2: Lesson 3 Lesson 6 Lessons 21-24
9-12.CT.c.4	Analyze a complex data set to answer a question or test a hypothesis (e.g., analyze a large set of weather or financial data to predict future patterns).	0: Lesson 5 2: Lessons 5-8 Lesson 11
9-12.CT.c.5	Identify different problems (e.g., large or multipart problems, problems that need specific expertise, problems that affect	0: Lesson 5 1: Lesson 1

	many constituents) that can benefit from collaboration when processing and analyzing data to develop new insights and knowledge.	2: Lesson 3 Lesson 5 Lessons 21-24 3: Lesson 1
9-12.CT.d.1	Use a development process in creating a computational artifact that leads to a minimum viable product and includes reflection, analysis, and iteration (e.g., a data-set analysis program for a science and engineering fair, capstone project that includes a program, term research project based on program data).	4: Lesson 19 Lessons 22-30
9-12.CT.d.2	Decompose a problem by defining functions, which accept parameters and produce return values.	4: Lesson 18 Lessons 22-26
9-12.CT.d.3	Select the appropriate data structure to represent information for a given problem (e.g., records, arrays, lists).	4: Lesson 19 Lesson 22-26
9-12.CT.d.4	Analyze trade-offs among multiple approaches to solve a given problem (e.g., space/time performance, maintainability, correctness, elegance).	1: Lesson 10 Lesson 20 4: Lessons 22-26
9-12.CT.d.5	Use appropriate looping structures in programs (e.g., FOR, WHILE, RECURSION).	4: Lessons 21 - 26
9-12.CT.d.6	Use appropriate conditional structures in programs (e.g., IF-THEN, IF-THEN-ELSE, SWITCH).	4: Lesson 15 Lesson 22-26
9-12.CT.d.7	Use a programming language or tool feature correctly to enforce operator precedence.	4: Lessons 22-26
9-12.CT.d.8	Use global and local scope appropriately in program design (e.g., for variables).	4: Lesson 16 Lessons 22-26
9-12.CT.d.9	Select and employ an appropriate component or library to facilitate programming solutions [e.g., turtle, Global Positioning System (GPS), statistics library].	4: Lessons 22-26
9-12.CT.d.10	Use an iterative design process, including learning from making mistakes, to gain a better understanding of the problem domain.	3: Lesson 9 Lesson 11 Lesson 24 4: Lesson 3 Lesson 7 Lesson 13
9-12.CT.d.11	Engage in systematic testing and debugging methods to ensure program correctness.	4: Lesson 27
9-12.CT.d.12	Demonstrate how to document a program so that others can understand its design and implementation.	3: Lesson 10 Lesson 22 Lesson 23 Lesson 25 4: Lesson 9 Lesson 11
9-12.CT.e.1	Create models and simulations to help formulate, test, and refine hypotheses.	2: Lesson 6
9-12.CT.e.2	Form a model from a hypothesis generated from research and run a simulation to collect and analyze data to test that hypothesis.	2: Lesson 6