**MVLA**
**2019-20**
**COURSE INFORMATION SHEET**

**Course Title:** Intro to Computer Science (BC1010)
**School:** MVHS
**UC/CSU requirement:** G Elective - 1 year required
**Textbook and/or other learning resources:** Shared Google Drive resources. The public resources and overall organization of the course can be found at the class website: **https://sites.google.com/a/mvla.net/mvhs-intro-to-cs/**. Some purchased curriculum components will only be accessible to students signed into an MVLA student Google account, while others are publicly accessible to all.

**Student Learning Outcomes:**

Students will learn the following computer programming concepts: abstraction, argument, ascii, binary, boolean, bug, character, code, command, conditional, constant, data types, debug, degrees, domain, evaluate, event, function, generalization, iterate, list, loop, mod (remainder), range, recursion, report, script, step, string, and variable.

Students will be introduced to these concepts using a variety of programming languages such as Snap!, Python, C, and Java. The Snap! language is based on drag-able commands and concepts -- such as "repeat until", "if...else", loops, lists, creating a function aka "block", defining variables, object oriented “sprites”, basic math computations, message passing, etc -- that are the foundation of written programming languages.

This course will then transition students to learn hard coding using several programming languages. The aim is for students to translate the logical concepts of programming blocks into written code. By working with a variety coding languages with different applications and levels of abstractions, students will thoroughly learn the core concepts of computer science and master coding fundamentals.

Students will specifically be able to:

1. Apply algorithmic, mathematical and scientific reasoning to a variety of computational problems
2. Design, correctly implement and document solutions to problems
3. Analyze and compare alternative solutions to computing problems
4. Work effectively in teams to design and implement solutions to computational problems

**Assessment and Grading (**BP 5121 / AR 5121**):** To ensure that every student has an equal opportunity to demonstrate their learning, the course instructors implement aligned grading practices and common assessments with the same frequency.

1. Grading categories and their percentage weights:

   40% - Projects
   35% - Classwork/Homework/Daily Log Files
   25% - Assessments/Quizzes

2. Achievement evidence collected within each grading category:

   Daily log files are graded on completion and quality. Quality is to be measured in two ways - the thoughtfulness and depth of their written reflection (composed of at least four complete sentences), and the quantity and quality of their warm-up/classwork coding assignments.

   Quizzes and projects are graded using rubrics that are given to students when the quiz/project is assigned.

3. Grading scales:

| | |
|---|---|
| 90-100% | A |
| 80-89.99% | B |
| 70-79.99% | C |
| 60-69.99% | D |
| 50-59.99% | F |

4. Homework/outside of class practices (AR 6154):

   Every day students are responsible for creating a daily log file that includes their programming/coding from the day, as well as written notes about what they worked on, where they left off and how it went. These daily log files must include the daily warm-up, any classwork assignments or ongoing project work, and a reflection composed of multiple complete sentences of the student's own writing.

   In general, there is little to no homework in this class. When a project deadline is given, students can definitely work on this outside of class time as needed to complete their project by the deadline.

5. Excused absence make up practices (Education Code 48205(b)):

   Upon returning from an excused absence, students must notify the teacher and work out a schedule to make up missed work. If the absence is planned ahead of time, the student must notify the teacher before leaving to work out a schedule. In general, a student is given as many days as they were absent to complete the assignment upon their return.

6. Academic integrity violation practices (MVHS Academic Integrity Policy):

   If a student violates the integrity policy (e.g. copy and pastes code from a friend or from the web) on a minor assignment, the student will earn a zero on the assignment, receive a warning, and the parent(s)/guardian(s) will be contacted.

   If a student violates the integrity policy on a major assignment (e.g. final semester project), the student will be referred to administration for further disciplinary action.

7. Late work practices:

   All work must be completed on time in order to receive full credit.

   Unexcused late work can be turned in within one week of the due date and will receive a maximum grade of 90%. If the student is absent on the due date, they have one week from the day that they return to school to complete the missed work. This applies to quizzes and projects.

   Daily log files cannot be turned in late, as they are a journal of the work done on that day, as well as how it went and any commentary on the experience.

8. Revision practices:

   Most every project and quiz in this class may be revised. Students have one week from the day the assignment grades are posted/returned to complete this revision. Revisions can receive a maximum grade of 90%.

   Daily log files cannot be revised, as these are a live record of what was accomplished on a specific day and how it went.

9. Extra credit practices:

   There is no extra credit in this course.

10. Additional grading practices:

    None

**Instructors' email addresses:**

brendan.dilloughery@mvla.net

**Additional information:**

The goal of this class is really three fold:

1. Have fun and relax - this is an elective you are choosing to take because you want to.
2. Solve problems/puzzles/challenges - every day there is a new challenge, and there isn't just one way to solve it. Talking with others about different solutions, learning from them, and comparing the pros and cons of different approaches is beautiful and fun and rich!
3. Learn the basics of programming in multiple languages. In this way, you learn the concepts and the logic. You can then more easily begin learning (or going deeper into) any programming language.