Center for Communication Technology Magnet – **Software Engineering 2**
Course Description and Expectations

Instructor: Mr. Fornstrom
Credit: 5 credit hours each semester
Prerequisite: 1) CCTM Software Engineering 1
Phone: 720-423-7164 **E-mail: stacey_fornstrom@dpsk12.org**
Website: http://www.sfornstrom.tjcctweb.com

INSTRUCTOR BIO

I have taught in the TJ Computer Magnet program for 17 years. I graduated from the University of Wyoming with a B.A. in Business Administration and from Texas A&M with a M.S. in Management Information Systems. Prior to teaching I worked as an accountant and a computer consultant for businesses.

COURSE DESCRIPTION

Software Engineering 2 is a continuation of Software Engineering 1. Completion of both semesters with a grade of "C" or better qualifies the student for **concurrent enrollment credit for CSC119** – **Introduction to Programming** at **Arapahoe Community College**. Beginning with the 2016-17 year, the College Board is offering a new course called **AP Computer Science Principles**. We will follow the AP Computer Science Principles curriculum so that Software Engineering 2 students will be prepared to pass the AP exam. The Computer Science Principles course is a great fit for CCT students because it emphasizes **problem solving** and **creativity.**

FEES

There is a $50 fee for this class for software, student supplies and membership in the **TJ Skills USA** club. Fees cover both semester 1 and semester 2. Please turn the money in to Mr. Fornstrom by **Friday September 8, 2017**. Checks should be made payable to **Thomas Jefferson High School**.

CLASS RULES AND PROCEDURES

All Thomas Jefferson school rules are followed. Please refer to the CCT Magnet class rules for details of the behavior and procedures followed in class.

MAKE-UP and LATE WORK

Excused Absence: The student will be allowed one day for each day absent, plus one extra day to make up work and tests missed.
Unexcused Absence: The student receives a grade of **0** for any work missed as a result of an unexcused absence.
Late Work: I realize there will be times when unforeseen circumstances (particularly with computers) make it difficult for work to be completed on time. If the student attended class but was unprepared with their assignment, they can turn the work in the next day for a maximum of 50% credit. No credit will be given for work turned in more than 1 day late. Please plan for the unexpected when completing assignments so that there is ample time to complete all work on time.

LAB & OFFICE HOURS

The computer lab is open every day from 7am to 3pm for student project work. Mr. Fornstrom is available in the lab from 7am to 3pm. Please contact him to schedule an appointment or additional lab time.

GRADING

Grades are based upon assignments, programs, quizzes, tests and participation points. There will also be several group projects. Each team member will earn both an individual grade and a group grade on each group project.

PARTICIPATION POINTS

In order to become proficient with computers, students need to **work and practice** with computers. To encourage this, 3 participation points are available each class day for being here, on-time, and on task. If a student has an excused absence they may come in and make-up the computer work to receive the day's participation points. Points lost for unexcused tardies or absences may not be made up. Points lost for inappropriate computer use (such as playing games or surfing the web during class time) may not be made up.

GRADING SCALE

90-100% = A
80-89% = B
70-79% = C
60-69% = D
Below 60% = F

Course Objectives and Curriculum Overview:

This course follows the College Board AP Computer Science Principles Framework. The main course objectives are summarized below in the 6 **computational thinking practices** and 7 **big ideas** for the course. For additional details of computational thinking practices and the big ideas, see the **AP Computer Science Principles and Exam Description**. The outline below is adapted from the **CodeHS syllabus**.

Computational Thinking Practices – represent important aspects of the work that computer scientists engage in.

- P1: Connecting Computing
  - Identify impacts of computing.
  - Describe connections between people and computing.
  - Explain connections between computing concepts.
- P2: Creating Computational Artifacts
  - Create an artifact with a practical, personal, or societal intent.
  - Select appropriate techniques to develop a computational artifact.
  - Use appropriate algorithmic and information management principles.
- P3: Abstracting
  - Explain how data, information, or knowledge is represented for computational use.
  - Explain how abstractions are used in computation or modeling.
  - Identify abstractions.
  - Describe modeling in a computational context.
- P4: Analyzing Problems and Artifacts
  - Evaluate a proposed solution to a problem.
  - Locate and correct errors.
  - Explain how an artifact functions.
  - Justify appropriateness and correctness of a solution, model, or artifact.
- P5: Communicating
  - Explain the meaning of a result in context.
  - Describe computation with accurate and precise language, notations, or visualizations.
  - Summarize the purpose of a computational artifact.
- P6: Collaborating
  - Collaborate with another student in solving a computational problem.
  - Collaborate with another student in producing an artifact.
  - Share the workload by providing individual contributions to an overall collaborative effort.
  - Foster a constructive, collaborative climate by resolving conflicts and facilitating the contributions of a partner or team member.
  - Exchange knowledge and feedback with a partner or team member.
  - Review and revise their work as needed to create a high-quality artifact.

Big Ideas - The seven big ideas of the course encompass foundational ideas of the field of computer science, and are denoted here by B1 through B7.

- B1: Creativity
  - How can a creative development process affect the creation of computational artifacts?
  - How can computing and the use of computational tools foster creative expression?
  - How can computing extend traditional forms of human expression and experience?
- B2: Abstraction
  - How are vastly different kinds of data, physical phenomena, and mathematical concepts represented on a computer?
  - How does abstraction help us in writing programs, creating computational artifacts, and solving problems?
  - How can computational models and simulations help generate new understanding and knowledge?
- B3: Data and Information
  - How can computation be employed to help people process data and information to gain insight and knowledge?
  - How can computation be employed to facilitate exploration and discovery when working with data?
  - What considerations and tradeoffs arise in the computational manipulation of data?
  - What opportunities do large data sets provide for solving problems and creating knowledge?
- B4: Algorithms
  - How are algorithms implemented and executed on computers and computational devices?
  - Why are some languages better than others when used to implement algorithms?
  - What kinds of problems are easy, what kinds are difficult, and what kinds are impossible to solve algorithmically?
  - How are algorithms evaluated?
- B5: Programming
  - How are programs developed to help people, organizations, or society solve problems?
  - How are programs used for creative expression, to satisfy personal curiosity, or to create new knowledge?
  - How do computer programs implement algorithms?
  - How does abstraction make the development of computer programs possible?
  - How do people develop and test computer programs?
  - Which mathematical and logical concepts are fundamental to computer programming?
- B6: The Internet
  - What is the Internet? How is it built? How does it function?
  - What aspects of the Internet's design and development have helped it scale and flourish?
  - How is cybersecurity impacting the ever-increasing number of Internet users?
- B7: Global Impact
  - How does computing enhance human communication, interaction, and cognition?
  - How does computing enable innovation?
  - What are some potential beneficial and harmful effects of computing?

Tentative Timeline:

We will not have a textbook, but instead use the on-line curriculum at **code.org** and **codehs.com.** This will allow students to work outside of the classroom and assure that curriculum is up-to-date. Below is a tentative timeline for the year.

Unit 1: Web Development (3 weeks) - 8/22 to 9/9
Unit 2: Introduction to Programming (4 weeks) - 9/12 to 10/7
Unit 3: Programming with JavaScript (6 weeks) – 10/10 to 11/18
Unit 4: Digital Information (6 weeks) – 11/28 to 12/21; 1/9 to 1/27
Unit 5: The Internet (6 weeks) – 1/30 to 3/10
Unit 6: Data (5 weeks) – 3/13 to 3/24; 4/4 to 4/21
Unit 7: Performance Tasks (6 weeks)
Unit 8: AP Exam Review (1 week)
Unit 9: Final Project (rest of year)

6-week end dates

1st semester: September 29, November 3, December 21
2nd semester: February 16, April 6, May 31