Course Title – Computer Programming I

Implement start year – 2014-2015

Revision Committee Members, email, extension –

Scot Butler sbutler@lrhsd.org ext. 8870
Chris Callinan ccallinan@lrhsd.org ext. 8364
Robert Kibler rkibler@lrhsd.org ext. 8583

Unit # 1, topic : Problem Solving Process

Students will be able to independently use their learning to model in a flow chart, pseudo code, and in actual code an algorithm to complete a specific task.

Stage 1 – Desired Results

Established Goals

2009 NJCCC Standard(s), Strand(s)/CPI #
(http://www.nj.gov/education/cccs/2009/final.htm)

Common Core Curriculum Standards for Math and English
(http://www.corestandards.org/)

NJ World Class Standards
Content Area: 21st Century Life and Careers
(http://www.state.nj.us/education/cccs/standards/9/9-4-K.htm)

9.4.12.K.66 Employ information management techniques and strategies to assist in decision-making

9.4.12.K.(3).8 Participate in a user-focused design and development process to produce Web-based and digital communication solution

9.4.12.K.(4).1 Identify and analyze customer software needs and requirements to guide programming and software development

9.4.12.K.(4).2 Create and use information technology strategies and projects plans when solving specific problems to deliver a product that

21st Century Themes
(www.21stcenturyskills.org)

☒ Global Awareness
☐ Financial, Economic, Business and Entrepreneurial Literacy
☐ Civic Literacy
☐ Health Literacy
☐ Environmental Literacy

21st Century Skills

Learning and Innovation Skills:
☒ Creativity and Innovation
☒ Critical Thinking and Problem Solving
☐ Communication and Collaboration

Information, Media and Technology Skills:
☒ Information Literacy
☐ Media Literacy
☒ ICT (Information, Communications and Technology) Literacy

Life and Career Skills:

meets customer specifications 9.4.12.K.(4).3 Identify and analyze system and software requirements to ensure maximum operating efficiency 9.4.12.K.(4).4 Demonstrate the effective use of software development tools to develop software applications 9.4.12.K.(4).5 Use the software development process to design a software and deliver it to the customer 9.4.12.(4).6 Produce a computer application, in code, to demonstrate proficiency in developing an application using the appropriate programming language	☒ Flexibility and Adaptability ☐ Initiative and Self-Direction ☐ Social and Cross-Cultural Skills ☒ Productivity and Accountability ☐ Leadership and Responsibility
Enduring Understandings: *Students will understand that . . .* *EU 1* • Students will understand that programs must be carefully be designed before they are written. *EU 2* • Students will understand that programmers use pseudo code and flowcharts to model a program EU 3 • Students will understand that programming languages have evolved to higher level languages EU 4 • Students will understand that algorithms have an end and a specified result EU 5 • Students will understand that computers store and read data using number systems based upon 0's and 1's	**Essential Questions:** *EU 1* • Why is it important to plan you program? *EU 2* • How can a flowchart be developed, read and interpreted? • Why are flowcharts important? EU 3 • How and why has programming evolved? EU 4 • What is the purpose of a data algorithm? EU 5 • How does a computer read and store data?

Knowledge: *Students will know . . .*	**Skills:** *Students will be able to . . .*
EU 1 • how to plan a program. *EU 2* • how to design, create and use a flow chart EU 3 • the differences between machine and high level languages EU 4 • the definition of an algorithm EU 5 • Binary, Octal, Decimal and Hexadecimal are the number systems used by computers	*EU 1* • plan a solution given a specific problem *EU 2* • Develop a flowchart that follows a plan EU 3 • explain why programming languages have evolved from Machine to a High Level Language EU 4 • Identify key aspects of a problem posed and develop an algorithm to solve the problem EU 5 • convert numbers between all of the computer number systems

Stage 2 – Assessment Evidence

Recommended Performance Tasks: *Each unit must have at least 1 Performance Task. Consider the GRASPS form.*

Other Recommended Evidence: *Tests, Quizzes, Prompts, Self-assessment, Observations, Dialogues, etc.*

- Flow charts including program flow and looping processes
- Pseudo code: a handwritten version of the program where the code is not written in full but rather an idea of the program flow is hashed out on paper
- Algorithms: Written code of the mathematical process that will allow data to be put into an array and sorted. The mathematical process of searching for data using different techniques.
- Test/quizzes
- Class discussion

Stage 3 – Learning Plan

Suggested Learning Activities to Include Differentiated Instruction and Interdisciplinary Connections: *Consider the WHERETO elements. Each learning activity listed must be accompanied by a learning goal of A= Acquiring basic knowledge and skills, M= Making meaning and/or a T= Transfer.*

- Convert numbers from any of the numbers styles to the one of the other three (M)
- Develop an algorithm for starting a car, starting with putting the key in the ignition (T)
- Develop an algorithm for determining a student's GPA given his/her credits and grade for each course (T)
- Convert algorithms into a flowchart (T)
- Desk-check the GPA algorithm (T)
- Debug an existing algorithm, provided by teacher, and convert to a flowchart (T)

The following is the suggested sequence of learning activities and number of days Computer Programming I class. (Approximate number of days 29)

- **Components of a computer system: Hardware, Software and Communications**
- **Evolution of computer Systems: History, Language Levels (Lower and Higher)**
- **Problem Solving Techniques: Problem Definition, Top Down Design**
- **Problem Solving Techniques: Flowcharting, Documentation and Debugging**
- **Program Design: Working with the user interface and the form designer**
- **Organization of Program Outcomes: Working with Interfaces**
- **Program Outputs: Message Boxes, List Boxes and output interfaces**

Vocabulary

- Programming Languages
- Machine Language
- High Level Language
- Object Orientated programming
- Documentation
- Code
- Binary
- Octal
- Hexadecimal
- Pseudo Code
- Algorithm