North Rose Wolcott Central School District

New Course Proposal

Submitting Teacher: Caitlin Wasielewski / Brian Read

Submission Date: 12.17.18 **Department/Grade Level:** 9-12

Course Title: Advanced Placement Computer Science A

Proposed Start Date: September 2019

Length: 1 year

Credit: 1.0

Prerequisite: Proficiency in Algebra (course & Regents Exam)

Enrollment: No cap

Staff needed (special qualifications if applicable): No additional staff needed

Course Description:

Per the AP website: AP Computer Science A is equivalent to a first-semester, college-level course in computer science. The course introduces students to computer science with fundamental topics that include problem solving, design strategies and methodologies, organization of data (data structures), approaches to processing data (algorithms), analysis of potential solutions, and the ethical and social implications of computing. The course emphasizes both object-oriented and imperative problem solving and design using Java language. These techniques represent proven approaches for developing solutions that can scale up from small, simple problems to large, complex problems.

Material or Special Equipment Needed:

Textbooks to be purchased through high school textbook funds

Summary of Alignment to the NYS Learning Standards:

Standard 5 for Technology Education in NYS is based on Computer Technology. One of the Commencement level standards is, "Students will develop an understanding of computer programming and attain some facility in writing computer programs."
The skills identified in the course description above -- problem solving, design strategies, organization of data, processing data -- can be applied in a variety of academic and authentic settings.

Course Justification:

See below

Please describe how this relates to the goals of the Strategic Action Plan:

We can justify the addition of this course because of its alignment with the Strategic Action Plan (SAP). The SAP requires us to increase the number of students taking advanced courses, to be responsive to our students' future story plans when identifying new course offerings, and to increase the number of advanced courses we offer. In recent years, we have had several students pursue post-secondary coursework in the field of engineering. We currently do not have many options for instruction in programming/coding, so this would be a welcomed addition for not only those students but also students who more broadly want to develop their problem solving and analytic skills in a technology-based context.

Please submit to Building Administrator by November 15th for review. All sections must be completed.

Next Steps:

- Building Administrator submits proposal to the ASI by November 25th.
- Leadership Council review proposal at the December LC meeting.
- Superintendent recommends new course to the BOE at the January BOE meeting.
  -Submitting teacher may be asked to attend the meeting to give information about the new course.
- Upon BOE approval, teacher will submit course outline/syllabus, including anticipated assessments used to the building principal, by May 1st.
- Counseling office will include the new course description in the Student Course Catalog.

This proposal will/will not be recommended for approval.

If not recommended for approval, please provide a reason:

Teacher may resubmit an amended proposal.

Building Administrator — Date: 12/18/18

Megan C Palioth
Assistant Superintendent for Instruction — Date: 1/23/19

AP Computer Science A

Topics Covered

Students will first be introduced to the basics of a computer and Java. While students do not need to be proficient in how and why a computer functions, they do need to understand the basic procedures in order to understand how and why computer programs work. This will also help them be more detailed in their programs, creating better programs in the long run.

Students will then move into a study of objects and data. This is where students will first be introduced to object oriented programing, the idea that a variable always points to something, such as an array, a list of numbers, or a word, instead of an action. Students need to learn the basic variables int (integers), double (rational numbers), and boolean (true/false statements). Students will also be introduced to the idea of strings (words). Students will learn to define the variables correctly and use them to create basic statements. We will learn to make computers and programs useful by programming them to do certain actions, such as arithmetic operators, or math expressions, as well as basic Java commands. Students will learn to deal with user input for the first time using textIO to read input, then making a program respond based on the input.

After this, students will learn about program statements such as the if loop, while loop, and for statement. These loops will allow students to write programs with more choice as well as more direct statements. Students will also learn to use pseudocode, or think through the steps of a program in regular English, then translate it to Java.

Once students understand the basics, they will be able to write their own somewhat developed class, or program. They will learn descriptors for classes such static, nonstatic, and void. They will then learn about methods, which lie within a class. They will also learn about the designing and testing model, which starts with writing a program, and extends all the way through debugging, or correcting, a program.

After this, students will shift their attention to arrays, which allow equal access to all members at the same time, meaning students can write concise programs. Students will learn about a one-dimensional array, as well as a two-dimensional array, which is considered an array of arrays. Students will learn about methods for searching arrays when elements are in order and not in order, as well as how to sort elements. Students will also learn about comparing array length and elements. We will also learn about the ArrayList Class and how to copy an array using the class.

Once students have a basic understanding of classes and objects, we can begin focusing on inheritance. Students will learn about the key is-a, meaning a parent to child relationship, and a has-a, meaning a child to parent relationship. We will learn about subclasses, including how to write and call a subclass. Students will also learn about inheritance and polymorphism. After this students will be able to apply what they learned to interfaces and their properties.

Students will learn about recursion and why it is useful in the programing world. They will learn about recursive programing and recursion in sorting.

Throughout the course, students will learn how to handle different exceptions, as well as the ethical and social implications of computer science and computer programming. The ultimate goal is for students to write correct, robust, and efficient programs.

Rationale

Our students would benefit from taking this course. Several of our recent graduates are studying fields in college where Computer Science is a required course, including Computer Science, Mechanical Engineering, and Software Engineering. Computer Science is foundational to the Computer Science field as well as many Engineering fields, which are all growing industries. Our students would benefit from taking Computer Science during high school and either learning the basics so they are more prepared for college, or earning college credits so they do not have to take the class in college.

There is student interest in this course. A small sampling of high school students were asked about whether a Computer Science course is something that would be interesting and beneficial to them, and some students were interested, especially students who want to go into technology or Engineering based fields after graduation.

This course also aligns with several steps in the Strategic Action Plan including:

2.2.4 Communicate with students regarding which AP courses could be offered, which would be most relevant given the path they are on post-HS, and assess student interest in which AP classes to offer

2.2.6 Expand advanced course offerings at the MS and HS

2.2.7 Increase the total percentage of students participating in advanced classes

Strategic Action Plan goal 2 is focused on increasing the number of advanced course offerings we have, as well as ensuring they are aligned to students future stories. Since our students are interested in fields based on Computer Science and fields that draw on Computer Science, this course is aligned to students future stories.

We do not currently offer any advanced Computer Science courses, or any courses that are based on Computer Science at all, so this would be a different type of advanced course that may encourage students to take an extra advanced course, or appeal to students who do not typically take advanced courses.

Example Programs

Students who take Computer Science A would be able to write programs that have comparable correctness, robustness, and efficiency to these examples by the end of the course:

This program asks a user to input the number of each type of coin they have, and the program will calculate the total amount of money they have.

Start Page · Stats.java · CountChangeWithScanner.java

Source | History

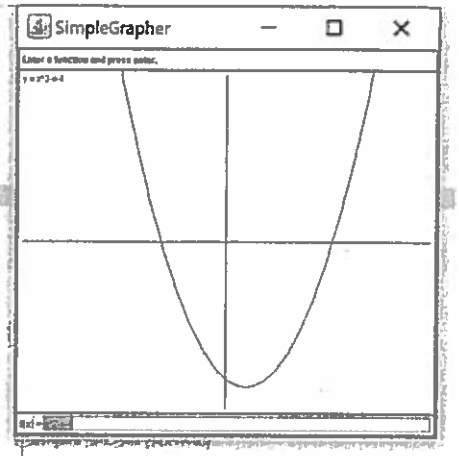
```java
13      public static void main(String[] args) {
14
15          int quarters;    // Quarters you have.
16          int dimes;       // Dimes you have.
17          int nickels;     // Nickels you have.
18          int pennies;     // Pennies you have.
19
20          Scanner stdio = new Scanner( System.in );
21
22          double dollars; // All the coins added together.
23
24
25          System.out.print("How many quarters do you have:  ");
26          quarters = stdio.nextInt();
27          stdio.nextLine();
28
29          System.out.print("How many dimes do you have:     ");
30          dimes = stdio.nextInt();
31          stdio.nextLine();
32
33          System.out.print("How many nickels do you have:   ");
34          nickels = stdio.nextInt();
35          stdio.nextLine();
36
37          System.out.print("How many pennies do you have:   ");
38          pennies = stdio.nextInt();
39          stdio.nextLine();
40
41
42          dollars = (quarters * .25) + (dimes * .1)
43                              + (nickels * .05) + (pennies * .01);
44
45
46          System.out.println();
47          System.out.print("Your total in dollars and cents is:");
48          System.out.printf( "%1.2f", dollars);
49          System.out.println();
```

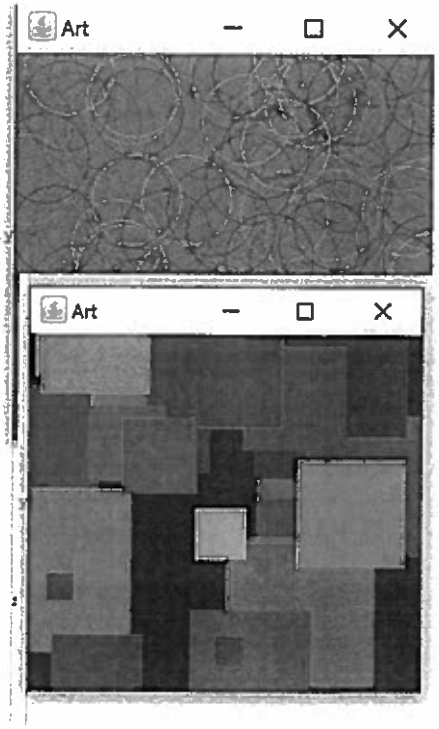
Output - CountChangeWithScanner (run)

```
run:
How many quarters do you have:  12
How many dimes do you have:     3
How many nickels do you have:   47
How many pennies do you have:   18

Your total in dollars and cents is:5.88
BUILD SUCCESSFUL (total time: 11 seconds
```

This program will graph a function the user puts in, similar to a graphing calculator.

```java
    private GraphPanel graph;  //The graph of the function.

    private JTextField functionBox;  //The box where the function is inputed.

    private JLabel status;  //The status of the program.

    public FunctionGrapher() {

        setBackground(Color.GRAY);
        setLayout(new BorderLayout(3,3));
        setBorder(BorderFactory.createLineBorder(Color.GRAY,3));

        graph = new GraphPanel();
        add(graph, BorderLayout.CENTER);

        status = new JLabel(" Enter a function and press enter.");
        status.setBackground(Color.WHITE);
        status.setForeground(Color.RED);
        status.setOpaque(true);
        status.setBorder( BorderFactory.createEmptyBorder(5,0,5,0) );
        add(status, BorderLayout.NORTH);

        functionBox = new JTextField();
        JPanel subpanel = new JPanel();
        subpanel.setLayout(new BorderLayout());
        subpanel.setBorder(BorderFactory.createEmptyBorder(3,3,3,3));
        subpanel.add(new JLabel("f(x) = "), BorderLayout.WEST);
        subpanel.add(functionBox, BorderLayout.CENTER);
        add(subpanel, BorderLayout.SOUTH);

        functionBox.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                Expr function;  //The function.
                try {
                    String def = functionBox.getText();
```



This program is similar to a screensaver on a computer. It randomly creates images of dots, lines, or boxes, and changes the picture after a specified amount of time.

```java
private class SquareArtData extends ArtData {
    class OneSquare {
        Color color;  // the color of the square
        int centerX;  // the center of square is (centerX, centerY)
        int centerY;
        int length;     // the length of a side of the square
    }
    ArrayList<OneSquare> squares = new ArrayList<OneSquare>();
    SquareArtData() {  // Constructor creates arrays and fills then randomly
        for (int i = 0; i < 25; i++) {
            OneSquare s = new OneSquare();
            s.centerX =  (int)(getWidth() * Math.random());
            s.centerY = (int)(getHeight() * Math.random());
            s.length = 30 + (int)(170*Math.random());
            s.color = new Color( (int)(256*Math.random()),
                    (int)(256*Math.random()), (int)(256*Math.random()) );
            squares.add(s);
        }
    }
    void draw(Graphics g) {  // Draw the picture.
        for ( OneSquare square : squares ) {
            g.setColor(square.color);
            g.fill3DRect(square.centerX - square.length/2, square.centerY -
                    square.length, square.length, true);
        }
    }
}
```