Intro to Computer Science - Programming Unit
By Jonathan Vanderah

Course Overview
This course will be an introduction to topics in the field of Computer Science, emphasizing security and ethics. We will spend time discussing topics covered in our FCCS course as well as introduce programming using Scratch. After this course, students will go on to take Computer Programming 1 where students will be introduced to both Python and Java.

The grading breakdown for the course is as follows: Coursework 20%, Quizzes 30%, Projects 30%, Final Exam 20%. Coursework is homework assignments, learning activities, and other minor tasks. Quizzes will be taken at the end of each unit for the Computer Science topics. Projects are the major assessments in this programming unit outlined in the Unit Outcomes. Dowling Catholic High School sets the percent for the Final Exam at 20%.

Unit Outcomes
● Outcome 1: Creating linear programs
  ○ Description: Students will be able to use the blocks in Scratch to create linear programs that reach a desired outcome.
  ○ Assessments: Students will be able to create stories in Scratch. First, they will recreate a single scene. Second, they will create an interactive story that changes based on user input.
  ○ Intermediate outcomes used: Movement, Broadcasts, User Input (text), Variables, Conditionals

● Outcome 2: Creating non-linear programs
  ○ Description: Students will be able to use the blocks in Scratch to create programs that use iteration and user input to reach a desired outcome.
  ○ Assessments: First, students will be able to create a game in Scratch that uses keyboard and/or mouse input to control a Sprite. Second, students will be able to make a game show program that utilizes data structures.
  ○ Intermediate outcomes used: Movement, Broadcasts, User Input (text), Variables, Conditionals, Iteration, User Input (Keyboard), Functions List

Intermediate Outcomes
● Movement
  ○ Description: Students will understand the X-Y axis, setting places when the program is started, and using movement blocks to control sprites.
○ Live-coding: I will demonstrate starting a new program in Scratch, using the “when green flag is pressed” block, and using various movement blocks in this program.
  ■ Students will be creating the program as I walk through the example. They will be encouraged to reflect, ask questions, and will be given opportunities to add and customize the code as we go.
  ■ Management of the activity will be having a program that covers all the blocks I want to demonstrate, monitoring engagement, and answering student questions.

○ Programming problem in class: Students will be given a background to put in Scratch that has four objects. The goal will be to produce code that moves the sprite to those four objects in a certain order, with different movements being used to get from one object to the next.
  ■ Students will work with a partner or individually. If working with a partner, they will be expected to discuss and compare movement options to each item and choose what works best for their group. If working individually, they will be expected to compare movement options and pick the ones that work best for them.
  ■ Management of the activity will be demonstrating adding a background to Scratch, explaining expectations for the activity and partner interaction, monitoring student progress and engagement, answering questions, and providing feedback to student work both during work and when they are completed.

● Broadcasts
  ○ Students will understand how to use the broadcast block to initialize actions and scripts in the program.
  ○ Live-coding: I will demonstrate creating a program that has code that starts when the flag is pressed, runs some code, and then broadcasts a message to trigger the next section of code. I will also demonstrate introducing a new sprite, adding sounds, changing a sprite’s costume, and changing the background.
    ■ Students will be creating the program as I walk through the example. They will be encouraged to reflect, ask questions, and will be given opportunities to add and customize the code as we go.
    ■ Management of the activity will be having a program that covers all the blocks I want to demonstrate, monitoring engagement, and answering student questions.
Programming problem in class: Students will be asked to pick a scene from a tv show or movie to create in Scratch. They will need to use 2 or more sprites, use broadcasts at least once, and either change a sprite's costume or the background.

- Students will be working on this individually. They will be expected to pick a scene, determine which sprites to use to represent the characters, determine which background(s) to use, and develop code to recreate the scene.
- Management of the activity will include laying out the requirements and expectations for the program, monitoring student engagement, answering questions, and providing feedback to student work both during work and when they are completed. Since they will be working individually, it would be ideal to check in with each student as they are working and give feedback at that time.

User Input (Text) and Variables
- Students will be able to take and use inputs to incorporate in their programs.
- Students will be able to store values and user inputs as variables.
- Live-coding: I will demonstrate creating variables, toggling if variable values are visible in the screen, and getting user text input to store and use in the code.

- Students will be creating the program as I walk through the example. They will be encouraged to reflect, ask questions, and will be given opportunities to add and customize the code as we go.
- Management of the activity will be having a program that covers all the blocks I want to demonstrate, monitoring engagement, and answering student questions.

Programming problem in class: Create a program where a sprite asks the user a set number of questions, and then uses theirs answers in the sprite's responses. Students will pick a theme for their questioning.

- Students will need to develop a theme, including sprites, background, and questions. They will be expected to complete this individually.
- Management of the activity will include laying out the requirements and expectations for the program, monitoring student engagement, answering questions, and providing feedback to student work both during work and when they are completed.
- **Conditionals**
  
  - Students will be able to use if/else blocks to make selections and control outcomes in their programs.
  
  - Live-coding: I will demonstrate using conditionals to make decisions as to what happens next in the program. This includes conditionals determining what a sprite says or does, layering conditional to create more than two outcomes, and using broadcasts in conditionals.
    
    - Students will be creating the program as I walk through the example. They will be encouraged to reflect, ask questions, and will be given opportunities to add and customize the code as we go.
    
    - Management of the activity will be having a program that covers all the blocks I want to demonstrate, monitoring engagement, and answering student questions.
  
  - Programming lab in class: There will be a step-by-step activity for the students to create a program that uses conditionals to respond to user input. It will be mostly scripted with a few opportunities for customization and will offer an option for extension for students who are excelling with the content.
    
    - Students will be working through this individually. They will be working through the activity and will be able to determine what customizations they will do (including if they do the extension or not)
    
    - Management includes outlining the lab and monitoring student engagement. I will check in with the progress of each student and give feedback. In that feedback, I will encourage students who are doing well to go for the extended challenge.

- **Iteration**
  
  - Students will be able to incorporate loops/iteration in order to repeat sections of code.
  
  - Live-coding: I will go over using the repeat and forever blocks, how to use conditionals to end forever loops, using loops to count or otherwise change variables, and when to use which loop.
    
    - Students will be creating the program as I walk through the example. They will be encouraged to reflect, ask questions, and will be given opportunities to add and customize the code as we go.
    
    - Management of the activity will be having a program that covers all the blocks I want to demonstrate, monitoring engagement, and answering student questions.
○ Programming problem in class: Students will work in groups of 2-3 on a problem that simulates ordering buffet meals at Pizza Ranch that will determine the total bill based on the ages of the customers.
  ■ Students will be working with their groups to determine the theming of sprites and backgrounds, how to set up the iteration, and how to set up their conditional statements inside the loop.
  ■ Management of this activity will include explaining the assignment and setting the prices and age ranges to be used. I will monitor the involvement of students in their groups and will monitor the progress of the groups. While checking progress I will give groups feedback on their work at that point.

● User Input (Keyboard for Sprite Movement)
  ○ Students will be able to create programs that use keyboard input to control sprites.
  ○ Live-coding: I will demonstrate how to use the events and sensing blocks for when keys are pressed to control a sprite. I will also go over using conditionals to check and limit a sprite’s x or y position, and how to use the “touching” block in conditionals.
    ■ Students will be creating the program as I walk through the example. They will be encouraged to reflect, ask questions, and will be given opportunities to add and customize the code as we go.
    ■ Management of the activity will be having a program that covers all the blocks I want to demonstrate, monitoring engagement, and answering student questions.
  ○ Programming lab in class: There will be a step-by-step activity for the students to create a maze program. It will be mostly scripted with a few opportunities for customization and will offer an option for extension for students who are excelling with the content.
    ■ Students will be working through this individually. They will be working through the activity and will be able to determine what customizations they will do (including if they do the extension or not)
    ■ Management includes outlining the lab and monitoring student engagement. I will check in with the progress of each student and give feedback. In that feedback, I will encourage students who are doing well to go for the extended challenge.

● Functions
- Students will be able to create their own function blocks to use in their programs.
- Live-coding: I will demonstrate how to create blocks in Scratch to use rather than repeating code. To do this, I will introduce using the pen feature and create shape functions to draw a picture.
  - Students will be creating the program as I walk through the example. They will be encouraged to reflect, ask questions, and will be given opportunities to add and customize the code as we go.
  - Management of the activity will be having a program that covers all the blocks I want to demonstrate, monitoring engagement, and answering student questions.
- Programming lab in class: Following instructions given to them, students will produce code that draws a certain picture.
  - Students will be working individually through this lab.
  - Management will be creating the outline of steps to create the picture, monitoring student engagement and progress, providing feedback, and answering student questions.
- Programming problem in class: Students will be selecting a picture to recreate in Scratch using functions. Their group will find a (school appropriate) picture of something they enjoy like a sport, a movie or tv show, or an activity.
  - Students will be working in groups of 2-3. They will need to brainstorm ideas of what to recreate, decide on a picture, and produce code to recreate the picture.
  - Management of this activity includes explaining the requirements and expectations, monitoring student engagement, checking in with the progress of each group and giving feedback, and answering student questions.

- Lists
  - Students will be able to create lists to store and reference data.
  - Live-coding: I will demonstrate creating lists and using lists in different situations. I will show getting items from a list, checking values in a list, and using multiple lists for questions and answers.
    - Students will be creating the program as I walk through the example. They will be encouraged to reflect, ask questions, and will be given opportunities to add and customize the code as we go.
Management of the activity will be having a program that covers all the blocks I want to demonstrate, monitoring engagement, and answering student questions.

- Programming problem in class: Students will be creating a program that simulates a quiz.
  - Students will be working either individually or in pairs. They will pick a subject for their quiz, create lists for questions and answers, and then create their program.
  - Management of this activity includes explaining requirements and expectations, monitoring student engagement, checking in on student progress, providing feedback, and answering questions.

Grading
Programming Problems and Programming Labs
1. These are the tasks students work on individually or in small groups after each intermediate outcome is covered.
2. These tasks have a set of requirements and a desired outcome for the program.
   a. The requirements are certain blocks that need to be use
   b. The desired outcome is a goal for the sprites or program to achieve, which can be reached using items beyond the requirements.
   c. These will be clearly outlined in task descriptions given to the students.
3. These tasks will be entered into the Coursebook section of the grade book.
   a. The other things in this category of the grade book will be any worksheets and reviews assigned in the Intro to Computer Science topics covered before Scratch.
4. This tasks will be graded on a 4 point scale:
   a. 4 - All requirements met and desired outcome achieved
   b. 3 - All requirements met but desired outcome not achieved OR Missing 1-2 requirement(s) but desired outcome achieved
   c. 2 - Missing up to half of requirements and desired outcome not achieved
   d. 1 - Missing almost all requirements and desired outcome not achieved
   e. 0 - Not turned in / Missing
5. Students will be able to make changes/redo/make-up these tasks to improve their score
6. No Late punishment in the grade book
7. Only way to get a zero is to not do it.
Assessments

1. There are four major assessments in this section of the course
   a. Two for each of the Unit Outcomes
2. Like the programming problems and labs, these will have a set of requirements
   a. There will be a guideline for desired outcomes as well, but these assessments are more open-ended
3. These will be in the Projects section in the grade book
   a. Kept separate from the Quizzes from the Intro to Computer Science sections due to differences in point totals
   b. Same weight in the grade book as those quizzes (30% each)
4. These will be graded out of 8 points. They will be broken down in the following categories:
   a. Program Description
      i. Students will be expected to begin their programs with a description of the program
      ii. This will vary based on the assessment
          1. For the story assessments, they will just need to introduce what story they are telling
          2. For the game and the game show, they will need to describe how to play and how to win
      iii. Scoring:
          1. 2 - Adequate description
          2. 1 - Description lacking details
          3. 0 - No description included
   b. Theming
      i. Students will be asked to have themes throughout their programs
      ii. This varies on the assessment
          1. For the story assessments, this is based on the fluidity of the story line.
          2. For the game, this is based on sprites and backgrounds used.
          3. For the game show, this is based on the questions asked.
      iii. Theme can also be included in the program description.
      iv. Scoring:
          1. 2 - Consistent theming throughout
          2. 1 - Theme not used consistently OR them introduced but then not used
          3. 0 - No noticeable theming
   c. Program Requirements
i. These assessments will have a set of requirements that must be used. Here are some examples in no particular order:
   1. Number of sprites
   2. Number of backgrounds
   3. Sounds
   4. Scoring
   5. Broadcasts
   6. Lists
   7. Loops
   8. Conditionals
   9. Keyboard input

ii. Scoring:
   1. 4 - All requirements met
   2. 3 - Missing 1-2 requirements
   3. 2 - At least half of requirements met
   4. 1 - Most requirements not met
   5. 0 - Not turned in / Missing

5. Students will be able to make changes and corrections to their programs to improve their score after submitting work.
6. No Late punishment in the grade book.
7. Only way to get a zero is to not do it.
8. I may end up converting these to a 4 point scale, but for now will leave out of 8.
   a. My students typically are confused if assessments are worth the same amount of points as other assignments. Even after explaining grading categories and weighted averages, students just assume more points for something means it is more important. So equal points means equal importance to them.

Discussion and Conclusion
Dowling is providing all students with a Chromebook for the next school year. In the past, I have had to teach programming classes in the business rooms that had 24 computers instead of in my own classroom that just had a set of 7 for students who needed them during work time. With having limited computers and not knowing what students have at home, I have traditionally tried to design the course so everything can be done in the classroom. I plan on keeping that the same as much as possible even though all students should be able to work outside of class with the Chromebooks.

A typical lesson for me starts with me going through a program explaining all the new material (blocks for Scratch, syntax and functions for other languages). After that, I have
1-2 more programs we create as a class where I am asking questions, having students provide ideas of what to do next, or having students try on their own before I go over how I would do it. Then once we finish examples, students begin to work on a task I have given them. These tasks are usually completion credit in the gradebook and students can redo them. Dowling wants us to give credit for tasks and activities in class beyond notes. These tasks have gone in the Coursework portion of the gradebook.

Grading is changing from what I have typically done coming from the math world. In our math classes, we would typically assign each question on an assessment a point value, tally up points, get their percent, and then put the test in the grade book out of 100 points (report the percent). This way all assessments were weighted equally. Quizzes and tests were in separate categories so tests were weighted more. Homework was always completion based and out of 10 points (also in their own category). We would always dock points if homework was turned in late, as Dowling’s policy is students can get at least half credit for late work. Some teachers just automatically go to half credit no matter how late. I always took 1 point off per day late, and I often would still give full credit based on the student’s situation.

Now that I am developing a new course that only I teach, I feel like I can start doing my own thing with grading and not just have to conform to a department. It is refreshing to apply my own beliefs as well as using new ideas from working with my peers in these courses. This grading is more like standards based, if you show me you can do it you get the credit. This is a more accurate picture of what students know, whereas our math grades always felt like they represented work ethic, effort, and if they can meet deadlines. Employability skills are important but shouldn’t have as big of an impact on the grade as content knowledge and ability.

The end goal of this unit is to introduce the basic structures and vocabulary of programming. Scratch is a great platform to introduce these topics as it is user friendly and students can visualize how programs need to be organized to produce an outcome. They don’t need to worry about learning syntax and symbolism, they can focus on structure and the fundamentals. This will set them up for success in the other programming course we offer.