Add Title Page here

Style Guide:

- learn to program
- Computer Science vs -computer science
- Use *italics* for vocab words included in the appendix

Introduction

In 2017 the Virginia Department of Education adopted new standards for computer science from kindergarten through high school. The standards are structured so that the K-8 standards shall be integrated into core instruction for all students. From middle through high school new standalone electives were also defined.

The challenge many educators experience with the computer science standards centers around the need to integrate. Schools and teachers must both learn the new computer science content, and wrestle with how to effectively integrate multiple subject areas.

This guide is intended to give teachers support in planning lessons that integrate computer science and core content. Most of the steps of lesson planning will be familiar: setting the learning objectives, planning learning activities, formative and summative assessments.

Unique to this guide is a focus on determining the level of computer science integration. At one end of the integration spectrum, topics covered by the computer science standards map almost completely to lessons currently taught at the elementary levels. With minor adjustments computer science can easily be injected into the lesson. An example would be reinforcing the word algorithm as classroom routines are taught. At the other end of the spectrum are lessons that need to cover computer science in a more in-depth manner. An example would be having students learn coding skills while reviewing science concepts like the Water Cycle.

Unlike planning lessons for single content areas, careful consideration must be given to designing activities to maximize the learning for both content areas being integrated. Things like:

- What new content is being introduced and what is being reviewed?
- How will vocabulary from different subjects be handled? What happens when the same word has different meanings in different subjects?
- What is the difference between a lesson using computer science to support new learning in a core content area, versus a lesson where new content is being delivered in both subjects?
- Will the content be assessed separately or together?

Careful consideration must also be given to the cognitive load, and the difficulty, of the new content being presented. If a new concept in the core subject area is known to be difficult for students to master, it may not be appropriate to introduce new content in computer science at the same time.

(* *Insert thanks to writing and review teams* *)

Defining Computer Science

Before exploring opportunities for integration educators need a solid understanding of what computer science is, and is not. Computer science moves beyond using technology tools towards an understanding of how they work and ultimately designing new solutions to enduring human problems. Despite common misperceptions, computer science is not simply programming. Like any scientific discipline there is a body of knowledge that informs how people understand and perceive the world around them, and practices for exploration, creation and experimentation. Programming, defined as giving computers instructions to follow, is a practice used in computer science. The field itself is much broader, much as biology is not simply conducting lab experiments.

The Association for Computing Machinery, in their 2003 report "A Model Curriculum for K--12 Computer Science: Final Report of the ACM K--12 Task Force Curriculum Committee" defined computer science as "the study of computers and algorithmic processes, including their principles, design, implementation, and impact on society" (Tucker, 2006, p. 2).

(*insert denning info on cs as a science? Or from book Untangling Complex Systems *)

Universal Human Questions

Across history, geography and culture there are core questions that all human societies have grappled with. These questions are universal and enduring, and people use the knowledge, systems and technology to tackle with them. As these questions are addressed new knowledge and technology is created. This cycle of discovery and application is at the core of the human experience.

Today humans have a vast array of tools and knowledge structures that drive this cycle. One primary example is the field of computer science. Whether through the application of programming to create and explore, or through the practice of computer science as a scientific endeavor, many of the drastic societal and technological changes humans have experienced in the new milleniea are driven by computer science.

So, what are these questions? In terms of elementary computer science this document focuses on __ core problems that influence what students learn in other core subjects.

(*third person or first? With out WE this sounds a bit stilted*)

- How can humans explore and understand the world around them? How can this knowledge be applied in new and creative ways?
- How can people express themselves?
- How do people stay safe?
- How can people communicate over distances? How can these communications be private? How do people know they arrived correctly?
- How can people do calculations more accurately?
- How can people collect and preserve information over time?

(*See other notebook to make sure this is all*)

Just as people in ancient China built the Great Wall to keep out ____, and the ancient Ancestral Pueblo peoples build towns on mesa walls for defensive reasons, now humans also need to worry about locking their front doors and and their personal information being stolen and abused. The field dedicated to online safety is cybersecurity.

Strands of Computer Science

At the K-12 level computer science is divided into the following strands: *Algorithms and Programming, Computing Systems, Cybersecurity (Virginia only), Data and Analysis, Impacts of Computing, and Networks and the Internet.*

Algorithms and Programming

This strand is at the core of how computer science gets done. Algorithms are step by step instructions that produce a result. In the context of computer science algorithms must:

- Use a common set of instructions that are clearly defined and produce consistent results.
    - In the context of programming these are the *commands*. They are precise and understood both by the human programmer, and the programming environment (like Scratch or Python) that translates them into commands the computer will run.
- The instructions are carried out in the correct order to produce the desired result.
    - Think about putting on a shoe before a sock, or ___example 2___. The steps may be correct and clear, but are not ordered to produce correct results..
- Produce a result and eventually end

Think about these steps in terms of a common algorithm taught in elementary school, long division. Would this algorithm work, or produce correct results, if the above features of an algorithm were not present?

Programming is then, the language used to implement algorithms on computers. It is how humans give instructions to the computer. It is helpful to remember computers are basically dumb boxes, despite all the buzz about artificial intelligence. They need exact instructions, in the right order, to produce correct results. All software used, from a web browser to an app on a phone, is written by a person or team of people using programming languages.

Programming languages are created by people and used for different purposes. For example Python is meant to be an easy to read language that can be used to create application for computers and on the web. On the other hand R was created specifically for doing statistical programming, and is often in the sciences to process data. These are both examples of text-based languages, where the commands are typed into the computer.

Another interface is the block based languages where the programmer drags blocks and puts them together in a specific order to create programs. There are several advantages at the elementary level, including the ability to quickly create interactive graphical programs and avoiding typos and spelling mistakes. While many of these were developed to be teaching tools for students, some university classes now also use them for computer science courses. Examples include Scratch, Code.org's AppLabb and StarLogo Nova for Project GUTS.

Computing Systems

This strand focuses on the hardware in computers. In computer science it is not just about memorizing components, but rather asking the fundamental question - what makes something a computer? As computers get smaller and invade more areas of our lives this has impacts on how students understand and experience the world around them.

For a device to be a computer it must include:

- Input - a way of translating information into a digital format that the computer can process. Examples include keyboards, microphones and cameras,
- Output - a way of translating the digital information computers process and store into a format humans can understand. Examples include screens, speakers and 3-D printers.
- Processor - the part of the machine that controls storing digital information and caries out the instructions. At the elementary level it is often compared to the human brain, it is the control cecter for everything the computer does.
- Memory - computers need things to process, this is stored in memory. On a mobile phone it might be pictures, and in a collaborative slide deck it might be stored in the cloud on a web server. The process of input and output are digitizing the world humans inhabit so it can be stored and processed, then changing it back to a format humans can understand. Everything stored in memory is stored numerically in binary.

Beyond indivisual devices -------systems

Cybersecurity

(*DEFINE *)

At the early elementary cybersecurity focuses on personal safety online. Students are asked to think about what their personally identifiable information is and how to protect it by using strong passwords.

(* Look up cybersecurity question from gal in C-Ville*)

Data and Analysis

At the core of computing lies data and analysis. The earliest computers

Impacts of Computing

Networks and the Internet

And in Virginia the Cybersecurity strad was pulled out from Internet and Networking and into its own *area.*

At the K-8 level wee think of this as

Math based - algorithms
Engineering lens - systems oriented - how things interact to create solutions

The Virginia Computer Science Standards of Learning

In 2016 the Virginia legislature passed HB 831 defining computer science and computational thinking, including computer coding, as core subjects for all students. In November of 2017 the State Board of Education adopted the Computer Science Standards of Learning.

The standards are based on two national documents, the Computer Science Teacher's Association Standards and the K-12 Computer Science Framework with a few key adjustments to meet needs in Virginia. First, while the national standards define five strands, Computing Systems, Networks and the Internet, Data and Analysis, Impacts of Computing and Algorithms and Programming, Virginia's standards add Cybersecurity as the sixth strand.

Secondly, while the national documents are organized into grade bands, the Virginia standards are defined for each grade level allowing more detailed definition of the expected outcomes for each grade level.

Finally, several suggested standards defined at the national level were removed or re-ordered so that all standards outside of programming tightly align with existing standards at each grade level.

The Computer Science SOLs are structured as follows:

- Kindergarten through eighth grade - the standards are integrated into core instruction
- A middle school elective is defined
- Three new high school electives are created

While there is a mandate to include computer science in kindergarten through eighth grade, the new electives for middle and high school are optional.

(*mention themes of each year in middle school ? Structure of modelling & simulation ?*)

The integrated computer science standards were designed specifically to facilitate integration. While the programming standards are new, all of the other strands are closely related to existing standards.

As an example, the first grade computer science standard 1.12 states *The student will identify and explain responsible behaviors associated with using information and technology*. This aligns to the History and Social Science standard 1.10 *The student will apply the traits of a good citizen by a) focusing on fair play, exhibiting good sportsmanship, helping others, and treating others with respect; b) recognizing the purpose of rules and practicing self-control*.

The final observation about the structure of the computer science standards relates to programming. The standards are intentionally designed to offer flexibility when introducing programming to students. In kindergarten and first grades the standards require students to program *either independently or collaboratively ... using a block based programming language or unplugged activities*.

In other words it is perfectly fine to do group activities off of computers where students design a set of step-by-step instructions to follow.

By second grade the language shifts to *both independently and collaboratively*, and in third grade they should be using a *block or text based programming language, both independently and collaboratively*.

So while programming can certainly be introduced earlier students do not need to be programming independently using computing devices until third grade.

Support Resources:

- Virginia Standards of Learning
- SOL Maps
- CodeVA
- Computer Science Framework
    - Defining Computer Science
-

Computer Science and Computational Thinking

CAN'T DO CT without a computer→ see denning

CT - the mental skills and practices for:

designing computations that get computers to do jobs for us, and

explaining and interpreting the world as a complex of information processes (Denning & Tedre, 2019)

CS is its own discipline - people get jobs as computer scientists

CT is a way of thinking and understanding - methodology

Computer science and computational thinking are deeply interrelated; both are concerned with algorithmic problem solving. Computational thinking is commonly defined as the set of thinking skills that when applied to real world problems create a solution that can be carried out by humans or machines. Thus, computational thinking is the bridge between the academic discipline of computer science and core content in K-8 classrooms. The practice of computational thinking distills the principles of computer science to a set of concepts and techniques that can be applied to any subject area.

Just like technology integration is not computer science, it is important to note that simply using a computer does not mean students are engaged in computational thinking. Computational thinking involves analyzing and decomposing problems to generate and refine algorithmic solutions. Computers may be essential tools for some computational thinking projects, but they can also be used for the passive intake of information. Central to designing lessons that integrate computer science is making careful decisions about when to incorporate technological tools to maximize student learning.

NOTES:

Computational thinking is about connecting computing to things in the real world.
Fred Martin http://advocate.csteachers.org/2018/02/17/rethinking-computational-thinking/

The standard definition is "Computational Thinking is the thought processes involved in formulating problems and
their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent"(Cuny, Snyder, & Wing, 2010; Aho, 2011; Lee, 2016).

Computer Science versus Technology Integration

Computer science is often confused with the use of computers and technology. In fact, computer science is its own academic discipline, built around a core body of knowledge and practices. At its center, computer science is "the study of computers and algorithmic processes, including their principles, their hardware and software designs, their [implementation], and their impact on society" (Tucker et. al, 2003, p. 6). Understanding computer science is of vital importance to students as computing technology is how we solve problems in the 21st century. It has a huge influence on how we live, work and play and students need a fundamental understanding of these impacts in order to navigate the modern world. Computer Science is used to create new technologies and to support exploration and discovery. **Mention creating, universal questions**

In contrast, technology Integration focuses on the use of existing technology to accomplish learning objectives. The goal is to pick technology tools that enhance learning. The student learning objectives remain focused on a particular curricular outcome, not the technology being used. This is distinct from integrating computer science, where there are curricular standards and outcomes for both the core subject area and computer science.

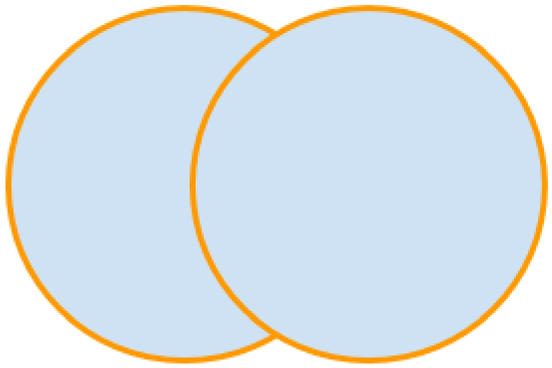
Thus, models for instructional technology integration may support computer science learning, but are often not enough for planning truly integrated computer science lessons. It is possible to create a computer science integrated lesson that is entirely unplugged and does not use any technological tools. Similarly, it is possible to build a computer science lesson and use the principles of technology integration to determine which device or platform best meets the learning needs of students. Either way, the focus is first on student outcomes then transitions to selecting appropriate learning activities and tools to maximize student learning.

Step 1: (* Integrating Content *)

Focus - how to select what to integrate

Venn diagram - new content

Goal - move students beyond being passive consumers of the things technology delivers and move them towards creating and expressing their own ideas, solving problems and exploring and ultimately steering (*their learning...too hookie??*)



Programming - Can be used for:

- Creative expression
- Presentation of information
- Modelling and simulation
- Storytelling
- Computation

Subject Areas

English

When we approach computer science as a literacy, not just a set of programming skills, it opens up a rich area of opportunity for integration between the two disciplines. The strands of English standards - Communication and Multimodal Literacies, Reading, Writing, and Research both inform how we can approach computer science topics, and uncover areas where computer science can enhance learning in English.

As an example, conveying information and ideas is a core area of overlap between computer science and English. The technologies created through computer science are really just solving ancient human problems. How do we share information over distances? At one time books were the new innovation that revolutionized how we disseminated stories and ideas. Today, we also use the Internet as a delivery method. The core human need to connect and be informed has not changed, just some of the ways we fill this need are new.

Thus, as we look for opportunities to integrate computer science and English there are two major areas of focus. First, computer science provides a rich toolset for students to use in expressing ideas, practicing writing and reading across different content areas. Second, computer science provides a lens of understanding for the traditional strands of English. It provides context for the subject in the 21st century (*HOW??*)

- **Communication**
  - The integration of computer science offers an opportunity for students to engage creatively as they prepare multimodal presentations. Unlike more static platforms, the incorporation of programming tools such as Scratch provide an opportunity to create presentations that are interactive and responsive to audience input.
  - Students must also develop an ability to analyze media messages. In an era where many of these messages are being delivered, and filtered, through social media this core literacy skill is intertwined with the Global Impact strand of computer science. Our viewing histories and searches are collected and aggregated and have a dramatic impact on the content we are then delivered. At the core, social media platforms deliver *us* as the product. Their entire purpose is to keep us engaged so that they can deliver advertisements their customers are paying for. There are many opportunities for students learn about media messaging, both as consumers and producers, to prepare them for the modern reality that technology creates.
  - The computer science global impact strand also asks students to explore the social and ethical issues that relate to computing devices and the networks. This is an opportunity for students to communicate and collaborate as they develop
- **Reading**
  - Too often in computer science we expect students to write code from scratch. Just like in English, where students are expected to both read and write, it is equally important to provide opportunities for them to read preexisting code. This is called *tracing* and it is a process of reading the commands and predicting what the program will do. These opportunities can be created in several ways. Buy providing pre-written programs for students to trace we model best practices in programming. We can also provide starter programs that students then modify and expand upon. Finally, students can read programs written by other students and give feedback for how to solve problems and improve the program. This process is especially motivating (*getting at they prefer each other as audience vs the adults - they care A LOT that the program does what they want it to do.(*conclusion*)
- **Writing**
  - Computers offer more than just embedded tools for correcting spelling and grammar. Computer Science also offers a systematic process for correcting and refining code called *debugging*. (the story of where the term *computer bug* comes from can be found here → possibly move to glossary). The goals of debugging parallel the goals for editing writing: correction of grammatical errors, modify language use for clarity and making adaptations to improve the final product. The process of debugging can both model how students may approach editing their work, and also help clarify the reasons why editing our work is important.
- **Research**
  - Both computer science and English emphasize the importance of recognizing the work other people have created and using correct citations.
  - Additionally, computer science and English both ask students to collect information and draw conclusions based on what they have found.

- Computer Science supports students ___ of citing other's original work. SOURCES,
- The student will give credit to sources when borrowing or changing ideas (e.g., using information, pictures created by others, using music created by others, remixing programming projects).

For each of the following include code and non-code areas that are "ripe" for integration

History and Social Studies

Language Arts

Math

Computer Science's origin is grounded in the field of mathematics. Historically humans have had the need to do large computations accurately. Given the very human tendency to make mistakes, and get bored and wander off after just a few minutes of doing computations this need has existed for millennia. And like many other innovations, computers are just the current tool we use to solve this age-old problem.

More than just a modern abacus, computers offer the ability to automate processes. Calculations? No problem. And by studying and advancing the tools (algorithms..)

Historic problem - calculate accurately and fast - this is what birthed CS in the beginning
Programming at its core is built around mathematical principles

Tools to enhance - ex: graphing X, Y coordinates
Core concepts that overlap

Mathematical Problem Solving
Mathematical Communication
Mathematical Reasoning
Mathematical Connections
Mathematical Representations

Science

Patterns of problems - loops vs taxonomies

CS supports exploration and understanding

Step 2: Identify Learning Outcomes

The first step is to decide the learning goals and outcomes for the lesson for all content areas.

The teaching goals are a good starting point. These are teacher focused, and include the skills, knowledge and understanding that will be taught in the lesson. In order to select learning activities that best meet these goals it is important to list them out for *all* content areas.

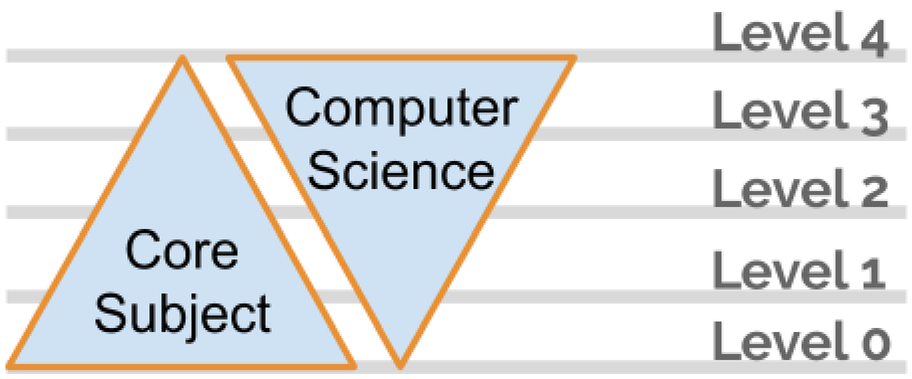
Once these are determined for all content areas, they can be made more specific through writing learning outcome statements in student-centered language, such as the students will be able to....

At this point the focus in not on developing activities, but keep the focus on what outcomes are for students. This will allow the selection of appropriate activities that maximize student learning.

Questions to Consider:

- What existing curriculum, pacing guides and standards exist? How will these influence the lesson goals and outcomes?
- Is there any content that may be confused because of integration? For example the term variable has different meanings in computer science, math and science.
- How will students know what subjects are being covered? (*this may be a learning activities*)

Step 3: Select level of Computer Science Integration



Level 0	No computer science content included in core content
Level 1	Computer science supports core content

Level 2	Computer science and core content coequal
Level 3	Core content supports new learning computer science
Level 4	Computer science content only

Level 0: Core content already covers computer science standards

Give examples (check in mapping doc)

There are several places in the K-8 standards

Level 1 Computer science supports core content

At this level the focus of the lesson is a core content area using computer science to support the learning.

Careful: new cs content should not be introduced here

Example: Students create a Scratch Jr. project to model the water cycle.

Questions to Consider:

- What strand of computer science will be included in this lesson?
- What computer science vocabulary will be needed for students to participate in this lesson?
- Will this lesson introduce new computer science content?
- If programming is to be included in the lesson, what specific programming skills will students need in order to be successful? (*link to Tools of Programming *) What supports will be offered for students lacking the required skills?
- Will the computer science concepts be a part of the assessment of the learning?
- (* something about how to handle it if students seek to use programming concepts beyond the minimum identified by the teacher *)

Level 2 Computer science and core content are coequal

Verbiage

Careful consideration should be given to new content being introduced.

Example: fsdf

Questions to Consider:

- Will this lesson introduce new computer science content? If so, how will this be presented? (*goal - will students know they are doing CS*)
- How will students balance the cognitive load of getting two new sets of content?
-

Level 3 Core content supports computer science

At this level computer science is the primary focus, with core content playing a supporting role. While new computer science content may be introduced, the core content being covered should be review of previous material.

Example:

Questions to Consider:

- Will this lesson introduce new core content?
- Will this lesson introduce new computer science content? If so, how will this be presented? (*goal - will students know they are doing CS*)

Level 4 Computer science content only

At this level, only computer science content is being covered.

Example:

Questions to Consider:

- How do these topics relate to core subjects? Are there opportunities to make connections to learning in core subjects

Step 4: Plan the lesson activities

This is the fun part - deciding on what

Inquiry, Equity and Content

Culturally relevant examples

Questions to consider:

- Is the computer science content best taught plugged or unplugged?
- What technology or materials will be needed for this lesson?
- How much time will the lesson take?
- What kind of formative and summative feedback activities are included?
- Are these activities accessible for all learners in the class?
- What questions can be used to move from “just the facts, m’am” to higher order thinking?
- Is there any content that may be confused because of integration? For example the term variable has different meanings in computer science, math and science. How will this….(*phrasing*)
- Is the level of integration appropriate? Does it need to be adjusted?

Appendices

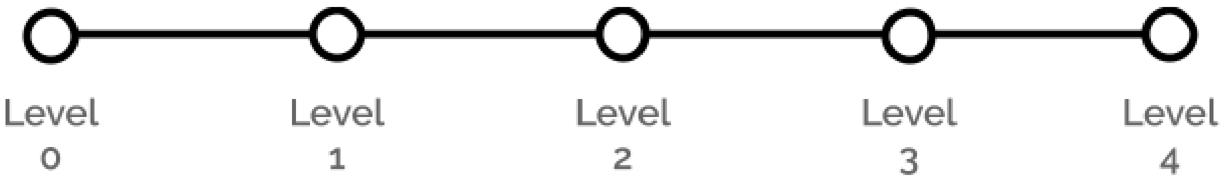
Appendix 1: Printable Integration Template

Step 1: What subject area(s) are you integrating? English Math Science Social Studies

Step 2: Select Learning Goals and Outcomes

Core Content Area:	**Computer Science Strand(s):**
Teaching Goals:	Teaching Goals:
Learning Outcomes - *student will…*	Learning Outcomes - *student will…*

Step 3: Select level of Computer Science Integration



Notes:

Step 4: Plan the lesson activities

Appendix 2: Glossary

computer science	
integration	askdjalkjd
plugged	
Unplugged	
Debug	
trace	

References and Resources

- Coyle, D. (2005). *CLIL: Panning tools for teachers*. University of Nottingham.
-

Using Anderson & Krathwohl's Taxonomy (2001) instead of Blooms

ECS - Joanna G's stuff

Understanding by design
Updated Bloom's Taxonomy
CLIL Toolkit

ASCED integration book
STEM Integration books - 2 one on kindle, other can be downloaded
Peter Denning books

A Model Curriculum for K--12 Computer Science: Final Report of the ACM K--12 Task Force Curriculum Committee - https://dl.acm.org/citation.cfm?id=2593247

Scratch, Code Applab, StarLogo Nova, Python

Thank to our team:

CodeVA staff:

Chris Dovi - executive director, and editor

Rebecca Dovi

Director of Education , CodeVA

Bryan Wallace

Title

(*Need name here*)

Natalie Rhodes

Title

TBD - Dee & Megan

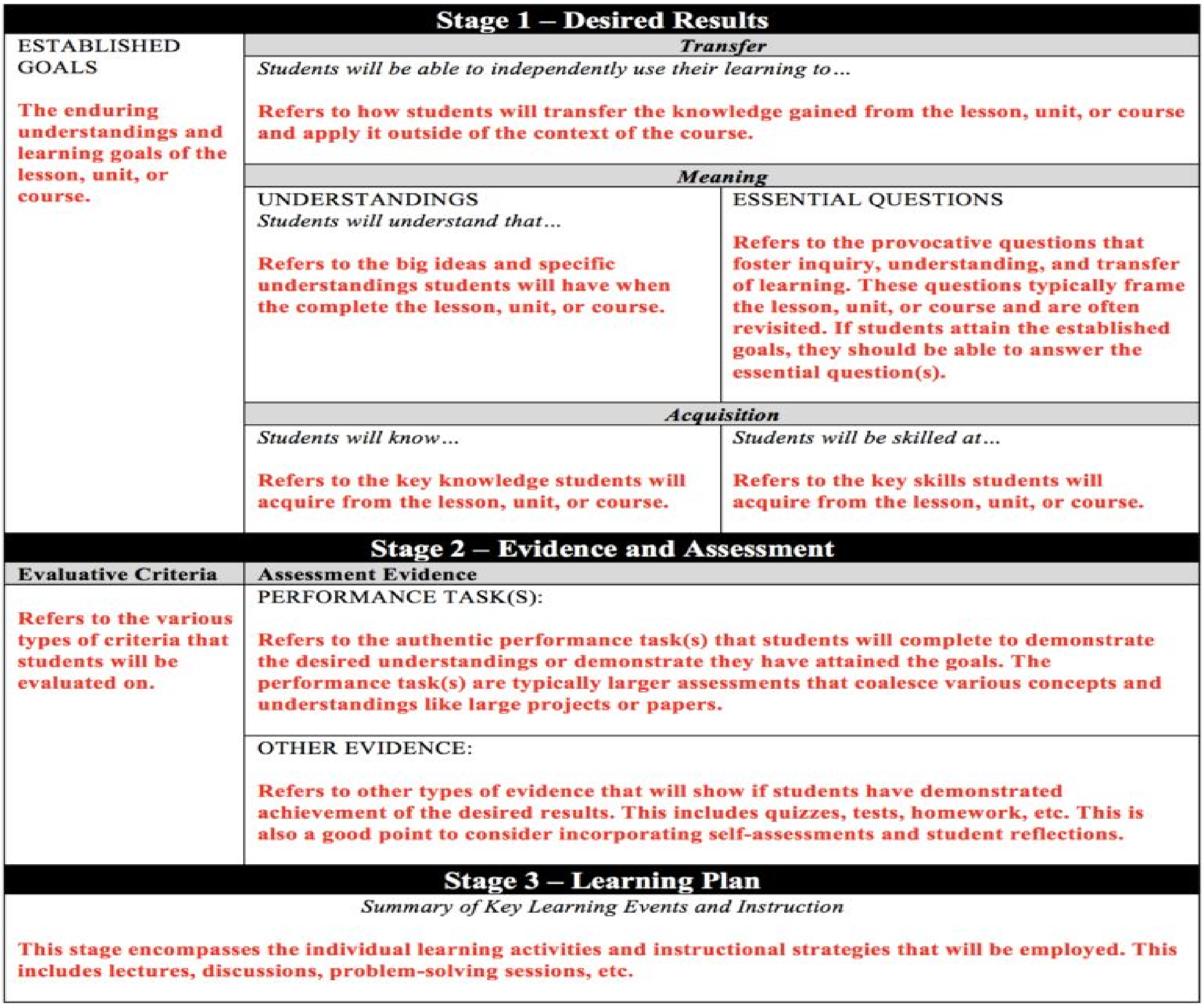
Review Panel:

TBD - Katie @ ODU
TBD - Todd Lash
TBD - molly and katie
TBD - chuck, tina, tim
TBD - Brylow - CS accuracy
TBD - gal in charlottesville - just that section

(any one else who review this)

Notes and Junk to erase:

Learning by design -- review and make adjustments



Stage 1 – Desired Results		
ESTABLISHED GOALS	***Transfer***	
The enduring understandings and learning goals of the lesson, unit, or course.	*Students will be able to independently use their learning to…* Refers to how students will transfer the knowledge gained from the lesson, unit, or course and apply it outside of the context of the course.	
	Meaning	
	UNDERSTANDINGS *Students will understand that…* Refers to the big ideas and specific understandings students will have when the complete the lesson, unit, or course.	ESSENTIAL QUESTIONS Refers to the provocative questions that foster inquiry, understanding, and transfer of learning. These questions typically frame the lesson, unit, or course and are often revisited. If students attain the established goals, they should be able to answer the essential question(s).
	Acquisition	
	Students will know… Refers to the key knowledge students will acquire from the lesson, unit, or course.	*Students will be skilled at…* Refers to the key skills students will acquire from the lesson, unit, or course.
Stage 2 – Evidence and Assessment		
Evaluative Criteria	Assessment Evidence	
Refers to the various types of criteria that students will be evaluated on.	PERFORMANCE TASK(S): Refers to the authentic performance task(s) that students will complete to demonstrate the desired understandings or demonstrate they have attained the goals. The performance task(s) are typically larger assessments that coalesce various concepts and understandings like large projects or papers.	
	OTHER EVIDENCE: Refers to other types of evidence that will show if students have demonstrated achievement of the desired results. This includes quizzes, tests, homework, etc. This is also a good point to consider incorporating self-assessments and student reflections.	
Stage 3 – Learning Plan		
Summary of Key Learning Events and Instruction This stage encompasses the individual learning activities and instructional strategies that will be employed. This includes lectures, discussions, problem-solving sessions, etc.		

Where does this fit into overall learning goals - what is the progression of content

Teaching aims vs student goals

- Which doc uses this (was it the CLIL framework??)
- https://thesecondprinciple.com/instructional-design/writing-curriculum/

Understanding by design:

Structure for this section

- Selecting broader learning aims
- Narrowing these down to more specific learning goals
- Eventually define Objectives - in **students will** format

Issue - balancing core content and computer science topics

Old Section 3

Determine the Cognitive Level

Begin with the individual objectives for the lesson from all content areas.

Note: the cognitive level for each content area may not match. As an example, if students are creating Scratch program to practice spelling words using conditionals in their code, the computer science content is at the create level, while the language content is at the remember level.

So using this lesson, the learning objectives could be:

- Objective 1: Students will create a program in Scratch to to practice spelling words. This includes demonstrating an ability to use conditional statements, voice recording and input and output in a program.
- Objective 2: Students will revise a program for correctness
- Objective 3: Students will correctly spell a set of vocabulary words.

Go through the taxonomy table (*include blank table in the appendix*) and place the objectives in the grid.

Knowledge Dimension	**Remember**	**Understand**	**Apply**	**Analyze**	**Evaluate**	**Create**
Factual Knowledge	Objective 3		Objective 1			
Conceptual Knowledge						Objective 1
Procedural Knowledge					Objective 2	Objective 1
Metacognitive Knowledge						

(*Do example grid*)

Chart - Bloom's Taxonomy Revised

(*this should be landscape in final PDF - on its own page - appendix?? *)

Remember	Understand	Apply	Analyze	Evaluate	Create
Retrieving relevant knowledge from long-term memory. ● Recognizing ● Recalling	Determining the meaning of instructional messages, including oral, written, and graphic communication ● Interpreting ● Exemplifying ● Classifying ● Summarizing ● Inferring ● Comparing ● Explaining	Carrying out or using a procedure in a given situation. ● Executing ● Implementing	Breaking material into its constituent parts and detecting how the parts relate to one another and to an overall structure or purpose. ● Differentiating ● Organizing ● Attributing	Making judgments based on criteria and standards. ● Checking ● Critiquing	Putting elements together to form a novel, coherent whole or make an original product. ● Generating ● Planning ● Producing

Anderson and Krathwohl's Revised Taxonomy - cite the article (Becca has it saved in Medeley)

Should they be picking different levels for each area of content?