美国中小学计算教育课程、路径和教师发展资源介绍

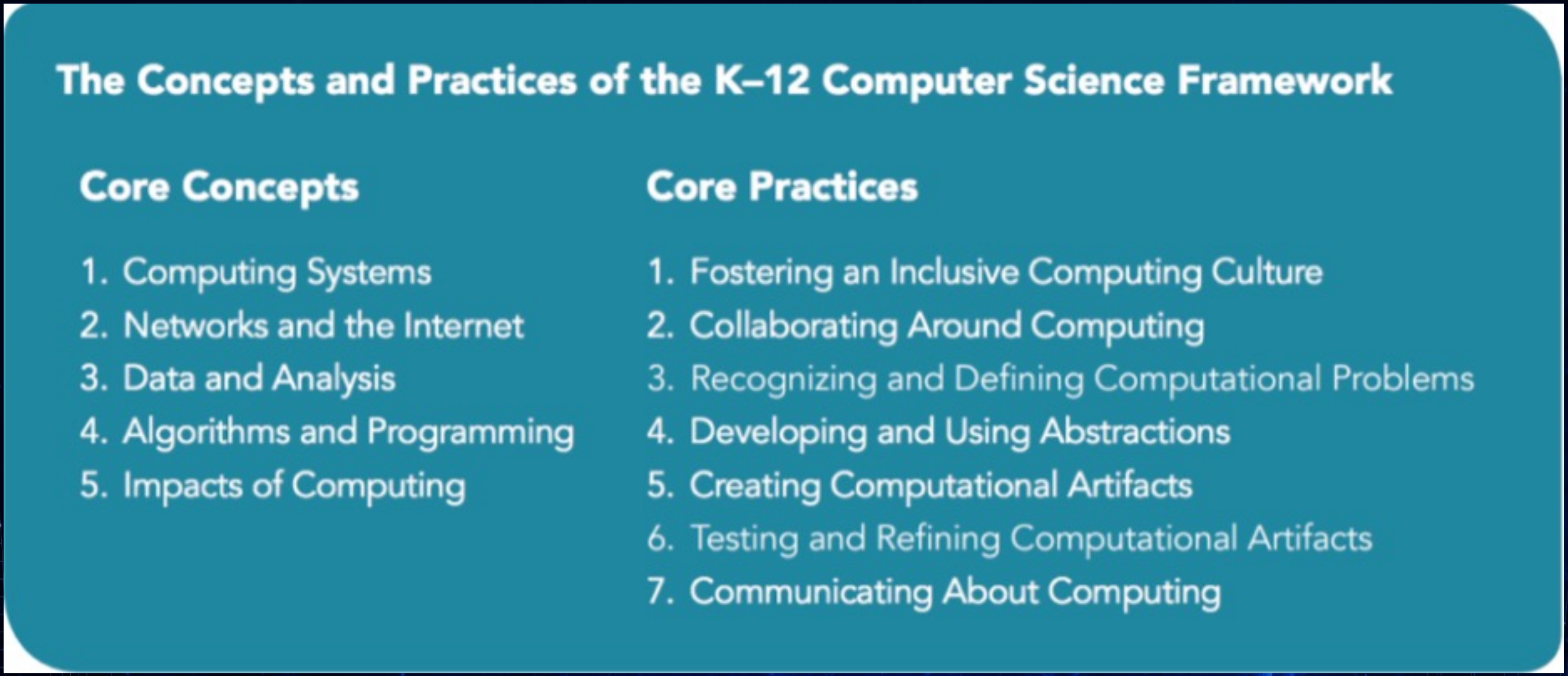
杨晓春

上海成趣信息科技有限公司

CNCC 讨论内容

- 1美国中小学计算机科学教育的框架的核心概念和核心实践
- 2美国小学、初中和高中各个阶段的计算机课程
  - ☆ 课程路径
  - ☆ 比较Scratch, Alice, 和SNAP！
- 3code.org组织和其他组织在中小学计算机教育中做的工作
- 4计算教育教师专业发展的方法和资料

美国中小学计算机教育框架的核心概念和核心实践



The Concepts and Practices of the K–12 Computer Science Framework

Core Concepts

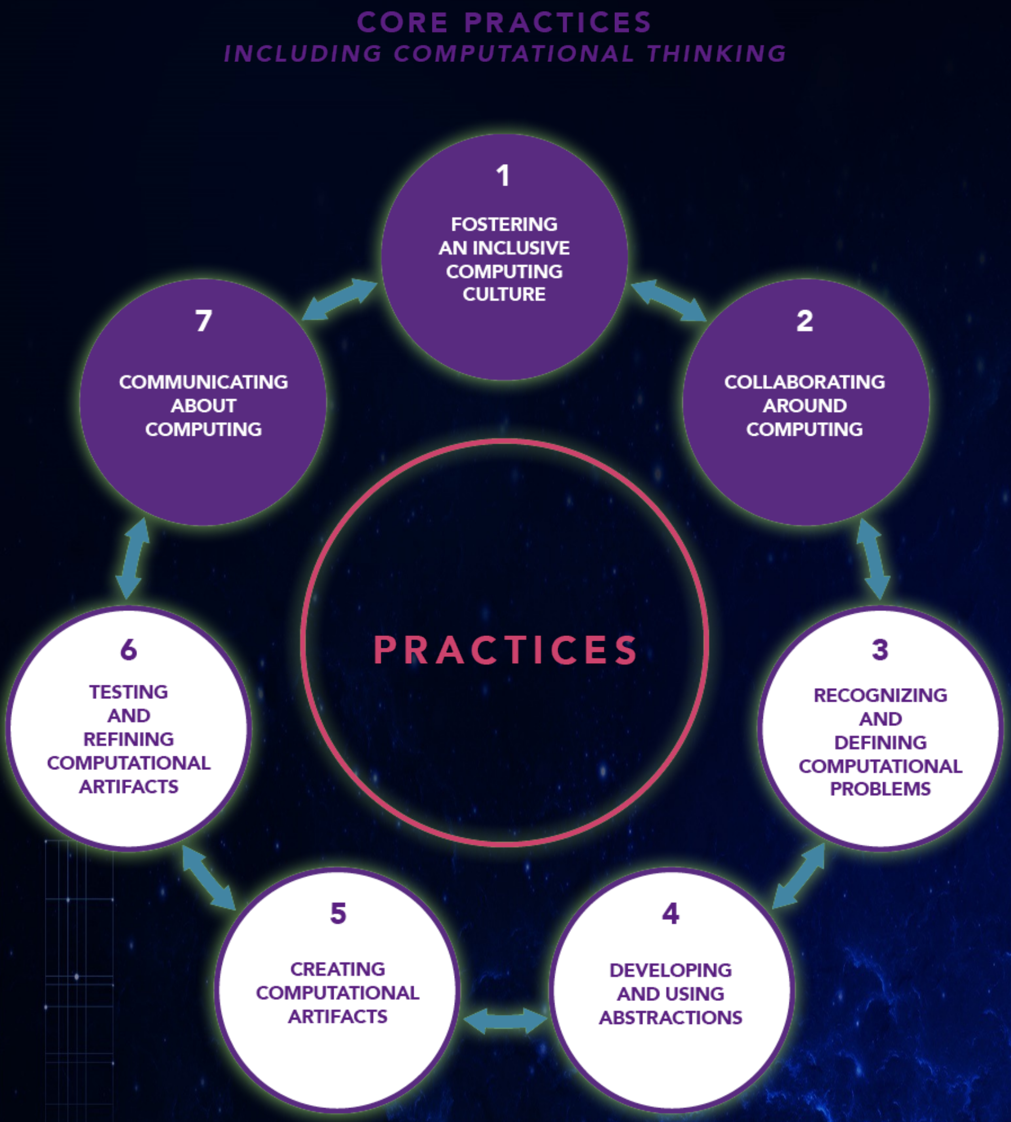
1. Computing Systems
2. Networks and the Internet
3. Data and Analysis
4. Algorithms and Programming
5. Impacts of Computing

Core Practices

1. Fostering an Inclusive Computing Culture
2. Collaborating Around Computing
3. Recognizing and Defining Computational Problems
4. Developing and Using Abstractions
5. Creating Computational Artifacts
6. Testing and Refining Computational Artifacts
7. Communicating About Computing

来源： https://k12cs.org/

CNCC 核心实践


CORE PRACTICES
INCLUDING COMPUTATIONAL THINKING
PRACTICES
1
FOSTERING AN INCLUSIVE COMPUTING CULTURE
2
COLLABORATING AROUND COMPUTING
3
RECOGNIZING AND DEFINING COMPUTATIONAL PROBLEMS
4
DEVELOPING AND USING ABSTRACTIONS
5
CREATING COMPUTATIONAL ARTIFACTS
6
TESTING AND REFINING COMPUTATIONAL ARTIFACTS
7
COMMUNICATING ABOUT COMPUTING

来源： https://k12cs.org/

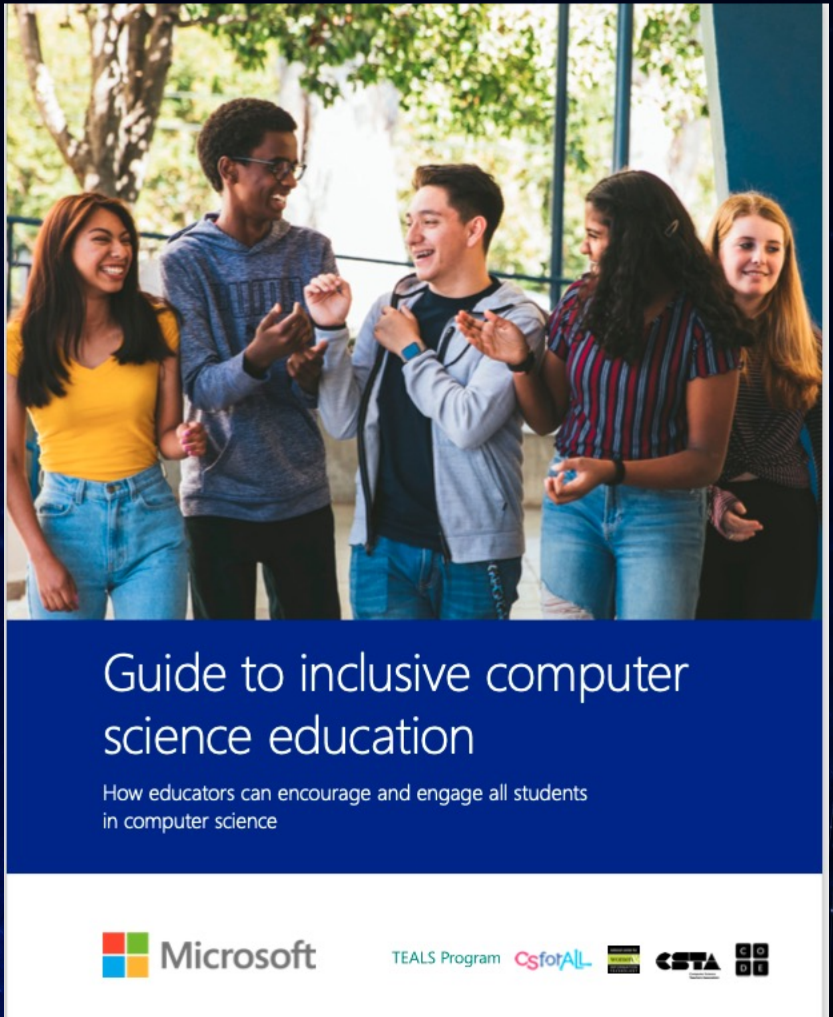
- 计算思维是计算机科学实践的核心，并被实践3-6所界定；
- 实践1、2和7是计算机科学中独立的一般实践，是对计算思维的补充；
- 计算思维指的是将解决方案表达为可由计算机执行的计算步骤或算法的思维过程；
- 计算思维要求理解计算机的能力，制定要由计算机解决的问题，并设计计算机可以执行的算法；

CNCC 计算教育教师专业发展的方法和资料

- 1美国中小学计算机科学教育的框架的核心概念和核心实践
- 2美国小学、初中和高中各个阶段的计算机课程
  - ☆ 课程路径
  - ☆ 比较Scratch, Alice, 和SNAP！
- 3code.org组织和其他组织在中小学计算教育中做的工作
- 4计算教育教师专业发展的方法和资料

美国小学、初中和高中各个阶段的计算机课程

- 包容性计算机科学教育指南，**Guide to Inclusive Computer Science Education**由 Microsoft、TEALS Program、CSforAll、NCWIT、CSTA 和 Code.org 于 2019 年创建
- 由于 CS 对许多学校和教师来说仍然是新事物，因此课程通常由一系列离散的技能和活动组成。 但就像数学、历史和其他科目一样，CS 最好作为一门累积科目来教授，其中每节课都建立在前一课的基础上



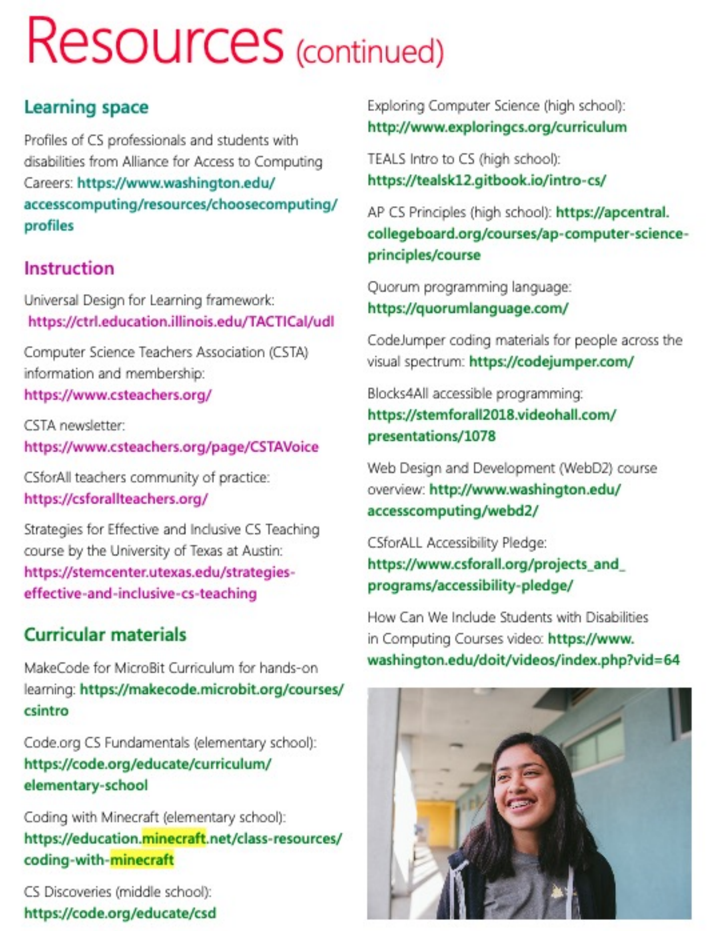
Guide to inclusive computer science education

How educators can encourage and engage all students in computer science

Microsoft TEALS Program CSforALL CSTA

美国小学、初中和高中各个阶段的计算机课程

- 渐进式 CS 课程：
- • 对于小学，请参阅 CS 基础和使用 Minecraft 进行编码；
- • 对于中学，探索 CS Discoveries 和 MakeCode；
- • 对于高中，请考虑探索计算机科学、TEALS 计算机科学入门和计算机科学原理先修课程



Resources (continued)

Learning space

Profiles of CS professionals and students with disabilities from Alliance for Access to Computing Careers: **https://www.washington.edu/accesscomputing/resources/choosecomputing/profiles**

Instruction

Universal Design for Learning framework: **https://ctrl.education.illinois.edu/TACTICal/udl**

Computer Science Teachers Association (CSTA) information and membership: **https://www.csteachers.org/**

CSTA newsletter: **https://www.csteachers.org/page/CSTAVoice**

CSforAll teachers community of practice: **https://csforallteachers.org/**

Strategies for Effective and Inclusive CS Teaching course by the University of Texas at Austin: **https://stemcenter.utexas.edu/strategies-effective-and-inclusive-cs-teaching**

Curricular materials

MakeCode for MicroBit Curriculum for hands-on learning: **https://makecode.microbit.org/courses/csintro**

Code.org CS Fundamentals (elementary school): **https://code.org/educate/curriculum/elementary-school**

Coding with Minecraft (elementary school): **https://education.minecraft.net/class-resources/coding-with-minecraft**

CS Discoveries (middle school): **https://code.org/educate/csd**

Exploring Computer Science (high school): **http://www.exploringcs.org/curriculum**

TEALS Intro to CS (high school): **https://tealsk12.gitbook.io/intro-cs/**

AP CS Principles (high school): **https://apcentral.collegeboard.org/courses/ap-computer-science-principles/course**

Quorum programming language: **https://quorumlanguage.com/**

CodeJumper coding materials for people across the visual spectrum: **https://codejumper.com/**

Blocks4All accessible programming: **https://stemforall2018.videohall.com/presentations/1078**

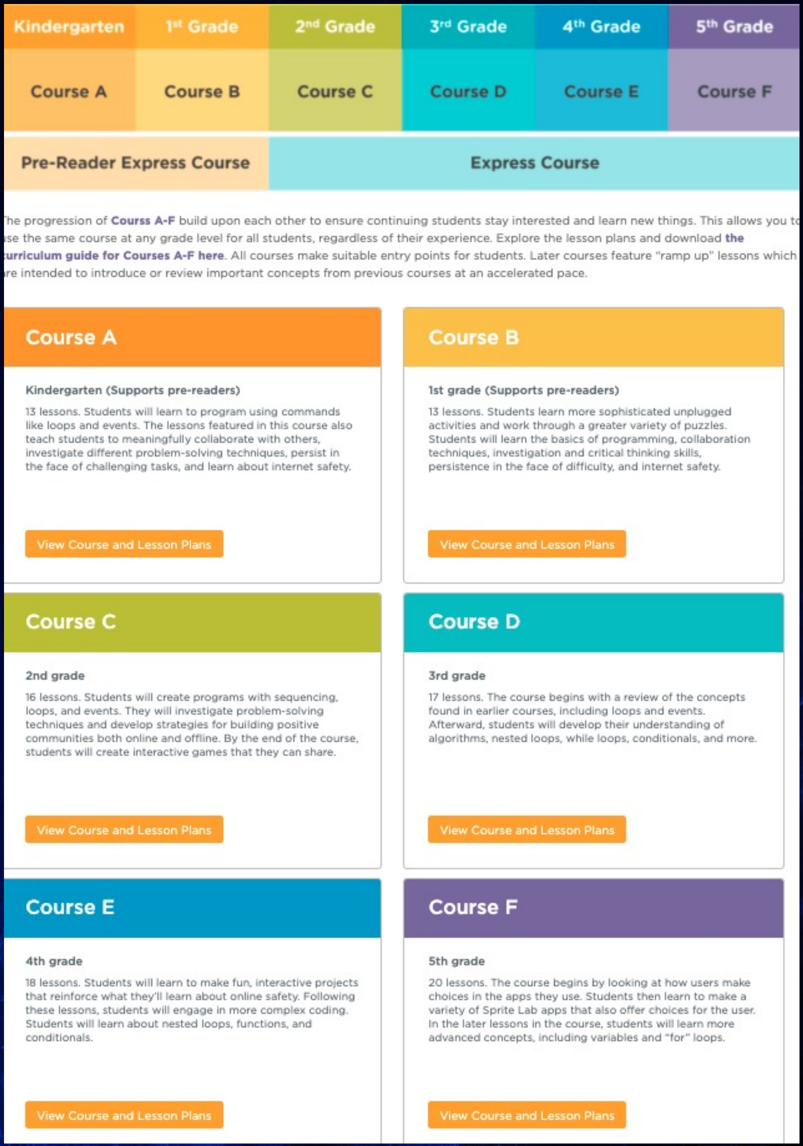
Web Design and Development (WebD2) course overview: **http://www.washington.edu/accesscomputing/webd2/**

CSforALL Accessibility Pledge: **https://www.csforall.org/projects_and_programs/accessibility-pledge/**

How Can We Include Students with Disabilities in Computing Courses video: **https://www.washington.edu/doit/videos/index.php?vid=64**

CNCC 小学课程介绍

- Code.org CS Fundamentals (elementary school):
  - **https://code.org/educate/curriculum/elementary-school**
- 六门计算机科学基础课程是为刚接触CS的教师灵活设计的，他们希望为他们的学生提供方便和公平的CS入门课程;
- 教授方法：作为课堂安排的一部分，每周的实验室或图书馆时间，支持数学和语言艺术的课程，或者制作创意项目;
- 促进者网络和区域合作伙伴在全国范围内提供专业的学习研讨会



Kindergarten	1st Grade	2nd Grade	3rd Grade	4th Grade	5th Grade
Course A	Course B	Course C	Course D	Course E	Course F
Pre-Reader Express Course		Express Course			

he progression of **Courss A-F** build upon each other to ensure continuing students stay interested and learn new things. This allows you t se the same course at any grade level for all students, regardless of their experience. Explore the lesson plans and download **the urriculum guide for Courses A-F here**. All courses make suitable entry points for students. Later courses feature "ramp up" lessons which re intended to introduce or review important concepts from previous courses at an accelerated pace.

Course A

Kindergarten (Supports pre-readers)

13 lessons. Students will learn to program using commands like loops and events. The lessons featured in this course also teach students to meaningfully collaborate with others, investigate different problem-solving techniques, persist in the face of challenging tasks, and learn about internet safety.

Course B

1st grade (Supports pre-readers)

13 lessons. Students learn more sophisticated unplugged activities and work through a greater variety of puzzles. Students will learn the basics of programming, collaboration techniques, investigation and critical thinking skills, persistence in the face of difficulty, and internet safety.

Course C

2nd grade

16 lessons. Students will create programs with sequencing, loops, and events. They will investigate problem-solving techniques and develop strategies for building positive communities both online and offline. By the end of the course, students will create interactive games that they can share.

Course D

3rd grade

17 lessons. The course begins with a review of the concepts found in earlier courses, including loops and events. Afterward, students will develop their understanding of algorithms, nested loops, while loops, conditionals, and more.

Course E

4th grade

18 lessons. Students will learn to make fun, interactive projects that reinforce what they'll learn about online safety. Following these lessons, students will engage in more complex coding. Students will learn about nested loops, functions, and conditionals.

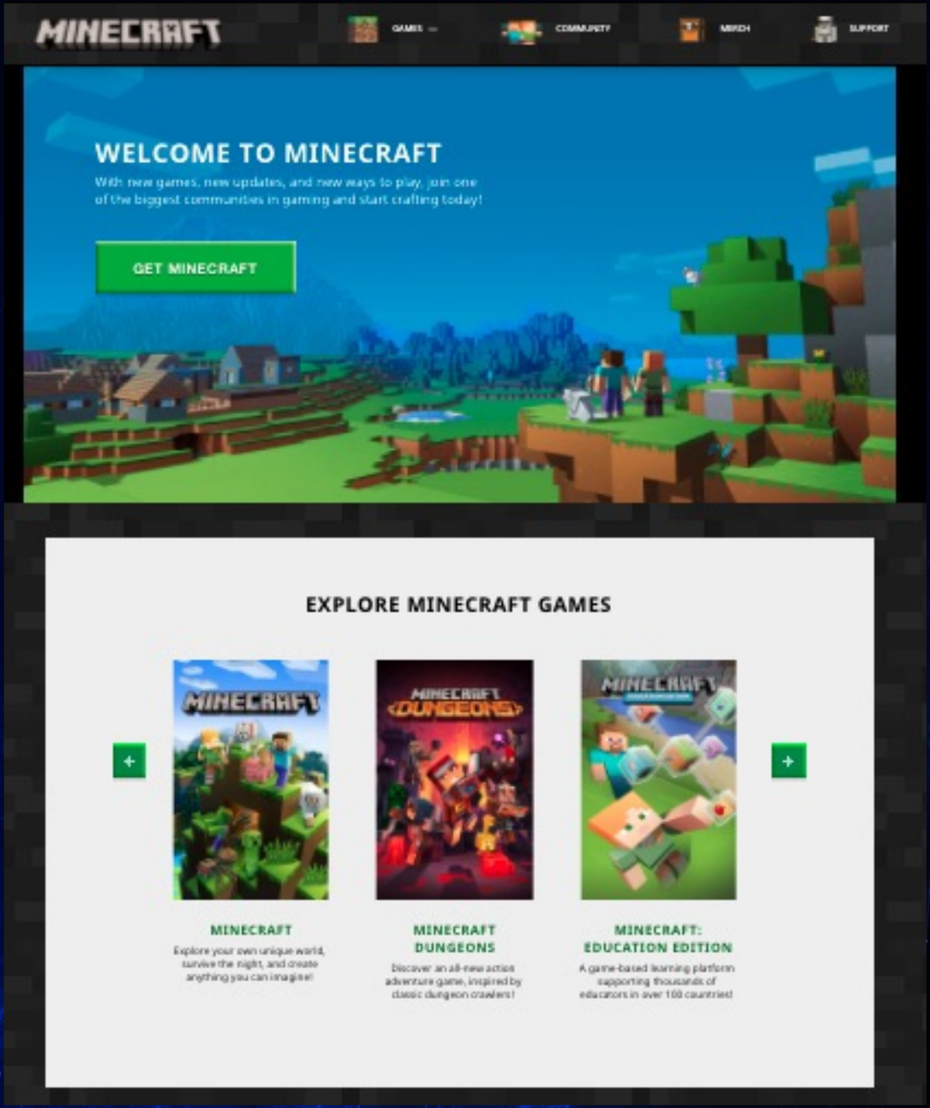
Course F

5th grade

20 lessons. The course begins by looking at how users make choices in the apps they use. Students then learn to make a variety of Sprite Lab apps that also offer choices for the user. In the later lessons in the course, students will learn more advanced concepts, including variables and "for" loops.

CNCC 小学课程介绍

- Coding with Minecraft (elementary school):
  - **https://education.minecraft.net/class-resources/coding-with-minecraft**
- 有了新的游戏、新的更新和新的玩法，请加入游戏中最大的社区之一，今天就开始制作吧!



WELCOME TO MINECRAFT

With new games, new updates, and new ways to play, join one of the biggest communities in gaming and start crafting today!

EXPLORE MINECRAFT GAMES

MINECRAFT

Explore your own unique world, survive the night, and create anything you can imagine!

MINECRAFT DUNGEONS

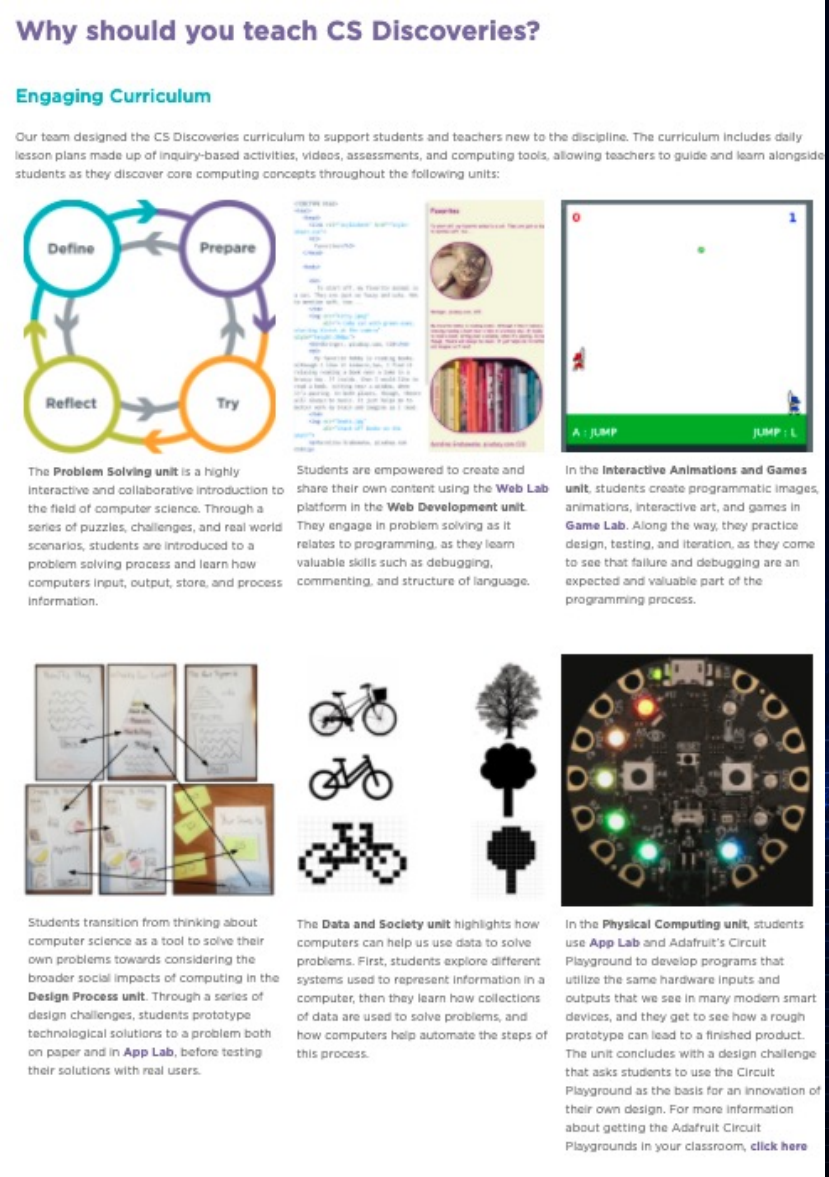
Discover an all-new action adventure game, inspired by classic dungeon crawlers!

MINECRAFT: EDUCATION EDITION

A game-based learning platform supporting thousands of educators in over 100 countries!

CNCC 初中课程介绍

- 探索 CS Discoveries 和 MakeCode
- CS Discoveries (middle school):
  - ☆ **https://code.org/educate/csd**



Why should you teach CS Discoveries?

Engaging Curriculum

Our team designed the CS Discoveries curriculum to support students and teachers new to the discipline. The curriculum includes daily lesson plans made up of inquiry-based activities, videos, assessments, and computing tools, allowing teachers to guide and learn alongside students as they discover core computing concepts throughout the following units:

The **Problem Solving unit** is a highly interactive and collaborative introduction to the field of computer science. Through a series of puzzles, challenges, and real world scenarios, students are introduced to a problem solving process and learn how computers input, output, store, and process information.

Students are empowered to create and share their own content using the **Web Lab** platform in the **Web Development unit**. They engage in problem solving as it relates to programming, as they learn valuable skills such as debugging, commenting, and structure of language.

In the **Interactive Animations and Games unit**, students create programmatic images, animations, interactive art, and games in **Game Lab**. Along the way, they practice design, testing, and iteration, as they come to see that failure and debugging are an expected and valuable part of the programming process.

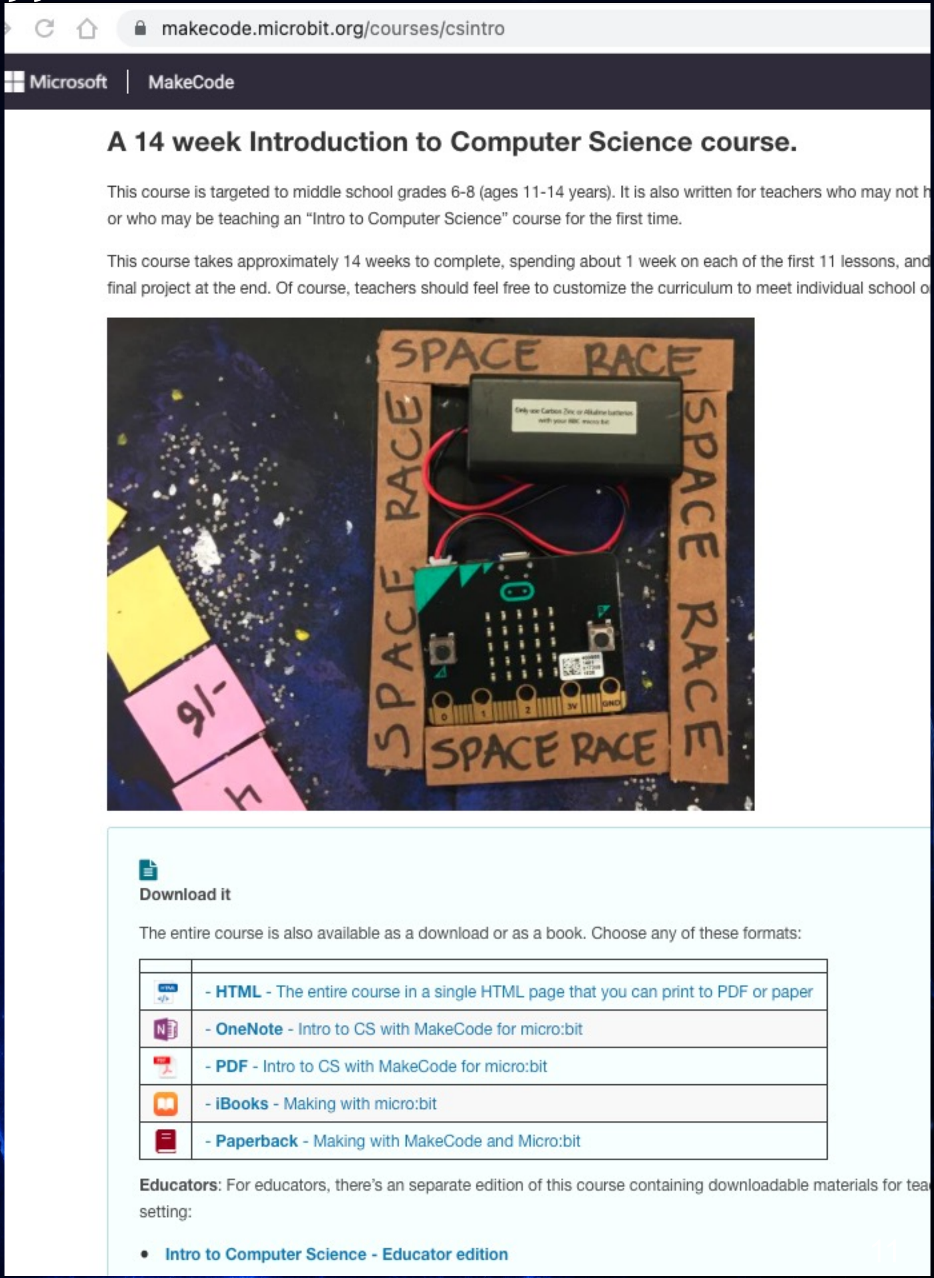
Students transition from thinking about computer science as a tool to solve their own problems towards considering the broader social impacts of computing in the **Design Process unit**. Through a series of design challenges, students prototype technological solutions to a problem both on paper and in **App Lab**, before testing their solutions with real users.

The **Data and Society unit** highlights how computers can help us use data to solve problems. First, students explore different systems used to represent information in a computer, then they learn how collections of data are used to solve problems, and how computers help automate the steps of this process.

In the **Physical Computing unit**, students use **App Lab** and Adafruit's Circuit Playground to develop programs that utilize the same hardware inputs and outputs that we see in many modern smart devices, and they get to see how a rough prototype can lead to a finished product. The unit concludes with a design challenge that asks students to use the Circuit Playground as the basis for an innovation of their own design. For more information about getting the Adafruit Circuit Playgrounds in your classroom, **click here**

CNCC 初中课程介绍

- MakeCode for MicroBit Curriculum for hands-on learning:
  - **https://makecode.microbit.org/courses/csintro**



A 14 week Introduction to Computer Science course.

This course is targeted to middle school grades 6-8 (ages 11-14 years). It is also written for teachers who may not h
or who may be teaching an "Intro to Computer Science" course for the first time.

This course takes approximately 14 weeks to complete, spending about 1 week on each of the first 11 lessons, and
final project at the end. Of course, teachers should feel free to customize the curriculum to meet individual school o

Download it

The entire course is also available as a download or as a book. Choose any of these formats:

	- **HTML** - The entire course in a single HTML page that you can print to PDF or paper
	- **OneNote** - Intro to CS with MakeCode for micro:bit
	- **PDF** - Intro to CS with MakeCode for micro:bit
	- **iBooks** - Making with micro:bit
	- **Paperback** - Making with MakeCode and Micro:bit

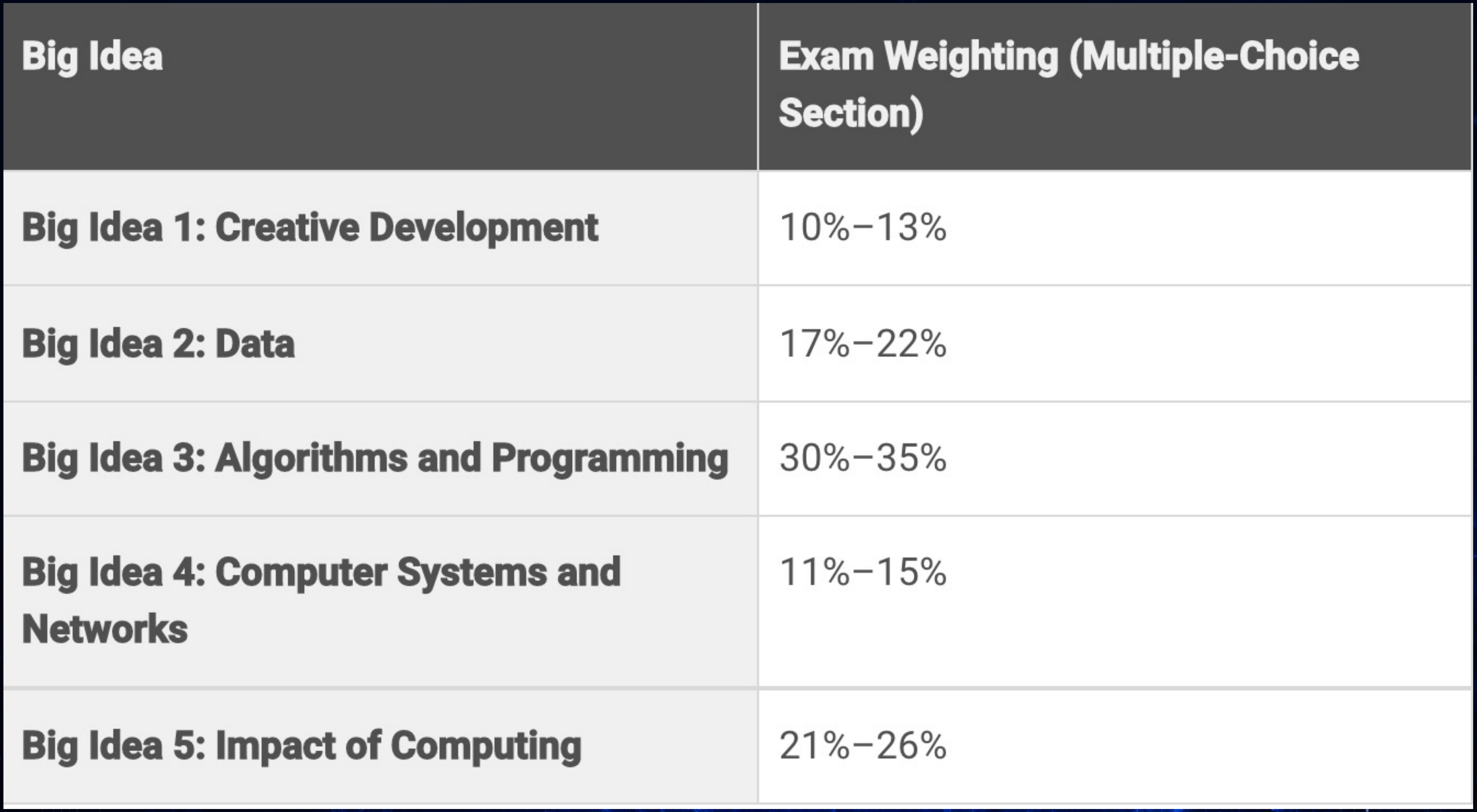
Educators: For educators, there's an separate edition of this course containing downloadable materials for tea
setting:

- **Intro to Computer Science - Educator edition**

CNCC 高中课程介绍

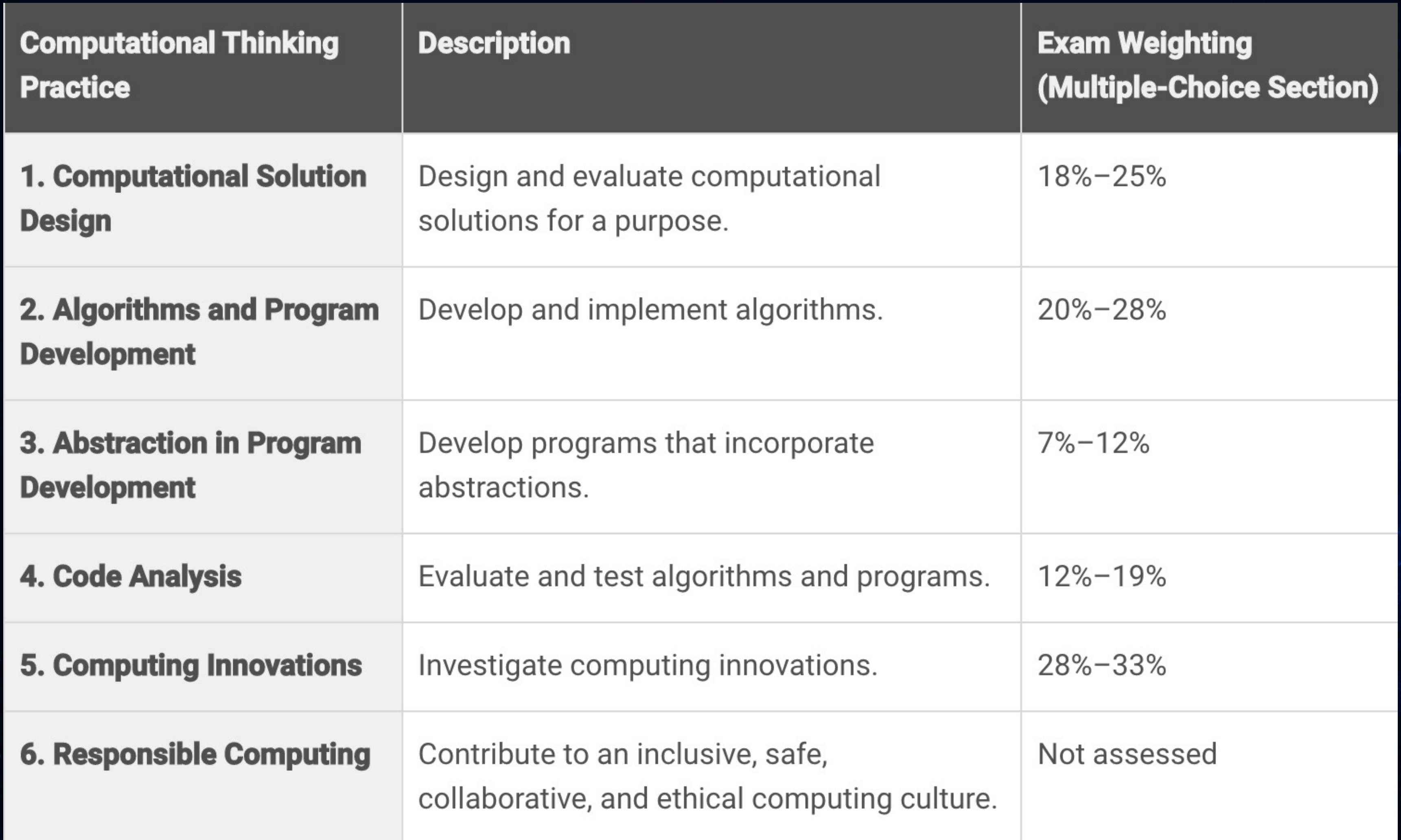
- 考虑探索计算机科学、TEALS 计算机科学入门和计算机原理先修课程
- Exploring Computer Science (high school): 由六个单元组成，每个单元大约六周，包括人机交互、问题解决、网页设计、程序设计入门、计算和数据分析以及机器人。还提供两个单元电子纺织品和人工智能，可以作为第5或第6单元的替代课程。
  - ☆ **http://www.exploringcs.org/curriculum**
- TEALS Intro to CS (high school):
  - ☆ **https://tealsk12.github.io/introduction-to-computer-science/**
- AP CS Principles (high school):
  - ☆ **https://apcentral. collegeboard.org/courses/ap-computer-science- principles/course**

高中课程介绍:计算机科学原理



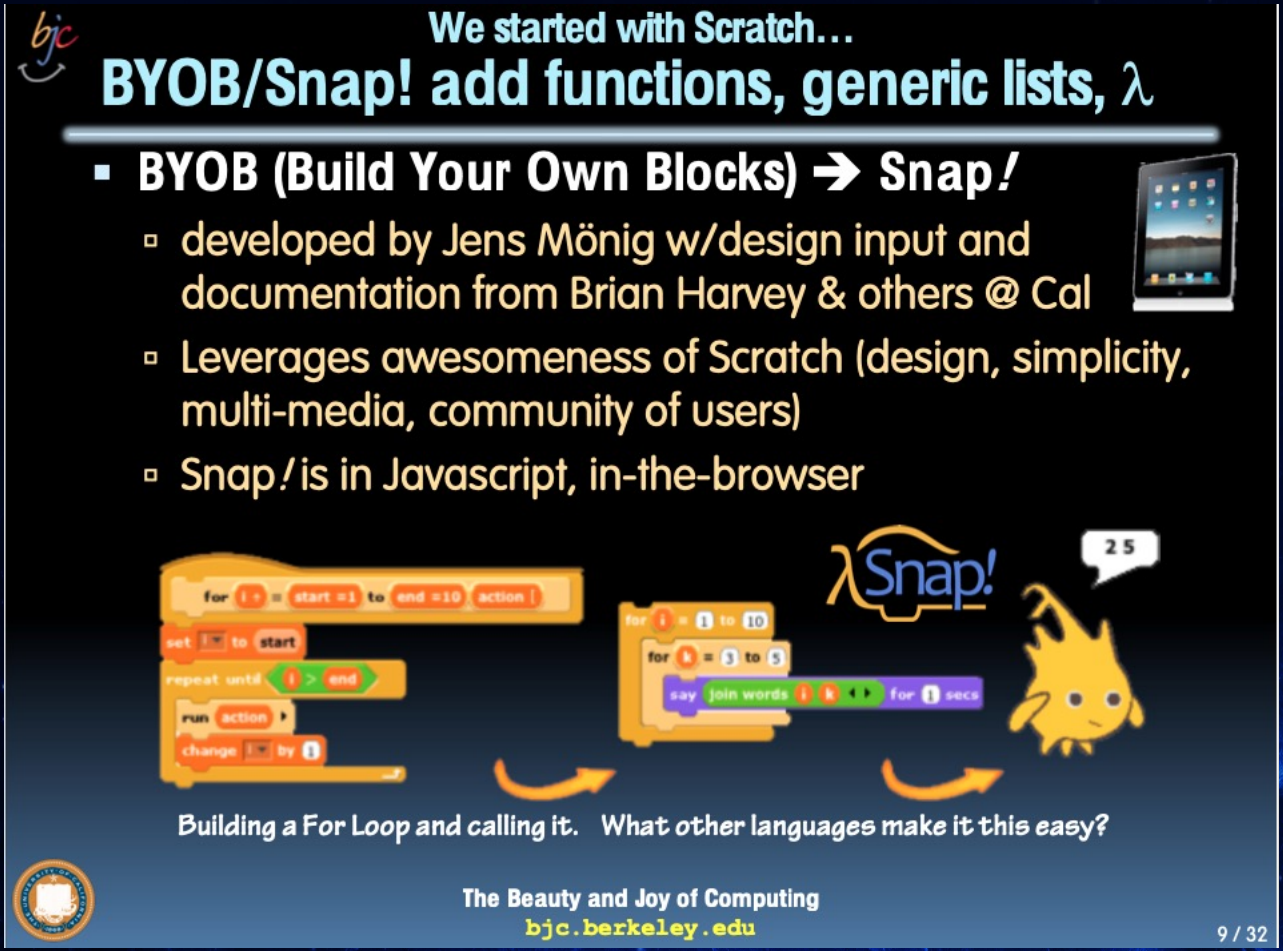
Big Idea	Exam Weighting (Multiple-Choice Section)
Big Idea 1: Creative Development	10%–13%
Big Idea 2: Data	17%–22%
Big Idea 3: Algorithms and Programming	30%–35%
Big Idea 4: Computer Systems and Networks	11%–15%
Big Idea 5: Impact of Computing	21%–26%

CNCC 高中课程介绍:计算机科学原理



Computational Thinking Practice	Description	Exam Weighting (Multiple-Choice Section)
1. Computational Solution Design	Design and evaluate computational solutions for a purpose.	18%–25%
2. Algorithms and Program Development	Develop and implement algorithms.	20%–28%
3. Abstraction in Program Development	Develop programs that incorporate abstractions.	7%–12%
4. Code Analysis	Evaluate and test algorithms and programs.	12%–19%
5. Computing Innovations	Investigate computing innovations.	28%–33%
6. Responsible Computing	Contribute to an inclusive, safe, collaborative, and ethical computing culture.	Not assessed

CNCC SNAP!介绍



We started with Scratch…

BYOB/Snap! add functions, generic lists, λ

- BYOB (Build Your Own Blocks) → Snap*!*
  - developed by Jens Mönig w/design input and documentation from Brian Harvey & others @ Cal
  - Leverages awesomeness of Scratch (design, simplicity, multi-media, community of users)
  - Snap*!* is in Javascript, in-the-browser

```
for (i) = (start =1) to (end =10) (action)
set [i] to (start)
repeat until <(i) > (end)>
  run (action)
  change [i] by (1)
```

```
for (i) = (1) to (10)
  for (k) = (3) to (5)
    say (join words (i) (k)) for (1) secs
```

Building a For Loop and calling it. What other languages make it this easy?

The Beauty and Joy of Computing
bjc.berkeley.edu

9 / 32

来源：Achieving CSforALL through The Beauty and Joy of Computing
UC Berkeley Teaching Professor Dan Garcia

CNCC 比较SCRACTH和SNAP!

- ✵ Scratch ：积木式图形化编程语言，可以创造互动故事、游戏和动画；https://scratch.mit.edu/ MIT支持；
- ✵ SNAP！(Build Your Own Blocks）：严肃的编程语言，是Scratch一个扩展的重新实现，具有第一类程序、第一类列表和第一类精灵的继承性，为了向青少年（高中到大学）教授计算机科学课程，Scratch面向小学到初中，https://snap.berkeley.edu/ UC Berkeley支持；
  - ☆ 技术特点：程序执行与开发环境整合；
  - ☆ First procedures第一类程序：可以使一个块或脚本成为一个变量的值，一个列表中的条目，等等；
  - ☆ First list第一类列表：列表可以有本身就是列表的条目；
  - ☆ 第一类：如果一种数据类型能够以与其他数据类型相同的方式使用，那么它在一种语言中就是第一类的。One of the slogans of Snap! is “Everything first class.”
  - ☆ 来源： https://snap.berkeley.edu/doc/Brian-Harvey_Baby-language.pdf “Why Do We Have to Learn This Baby Language?” Brian Harvey, Teaching Professor Emeritus, University of California, Berkeley

CNCC ALICE

- ALICE：3维互动动画程序可视化环境
- http://www.alice.org/ CMU，Oracle支持
- 是一个创新的基于积木的编程环境，可以轻松地创建动画，建立互动叙事，或在3D中编写简单游戏。与许多基于拼图的编码应用程序不同，Alice通过创造性的探索来激励学习。
- 旨在教授逻辑和计算思维技能、编程的基本原则，并成为面向对象编程的首次接触。
- 提供了补充工具和材料，这些工具和材料涉及不同的年龄段和主题，在吸引和留住计算机科学教育中的不同群体和服务不足的群体方面有明显的好处。

CNCC SCRATCH和ALICE

- 作用：1）教学生如何使用代码；2）创建教育游戏，进一步促进学生的学习；
- Scratch比Alice更简单，更容易理解。对于没有或很少有编程经验的学生来说，Scratch是最好的开始，对于像Alice这样更复杂的程序来说，它是一个很好的“入门”；
- 与Alice相比，Scratch的一个优势是它能够直接上传到Scratch的在线论坛。Scratch有一个“分享”按钮，可以直接将新制作的动画/游戏上传到Scratch网站上，设计者可以从同行那里得到反馈和建议，使他们的项目变得更好。Alice没有提供这个选项；
- 就图形和易用性而言，Scratch更容易使用，尽管它看起来没有Alice那么“成人”。Alice具有实际的专业视频游戏的外观和感觉，而Scratch显然更可爱，而且是为年轻观众准备的。Scratch作为人文/写作工具的实用性，Scratch被用于各种类型的诗歌和歌曲的创意展示，以及讲故事。
- 如果想把一些计算机科学融入现有的数学和代数课程中（无论是初中还是高中），建议初中生使用Scratch，而高中生则使用Scratch-Alice的顺序。
- 来源：https://sites.google.com/site/lqwsunrise/alice/scratch-vs-alice

CNCC 计算教育教师专业发展的方法和资料

- ✵ 1美国中小学计算机科学教育的框架的核心概念和核心实践
- ✵ 2美国小学、初中和高中各个阶段的计算机课程
  - ☆ 课程路径
  - ☆ 比较Scratch, Alice, 和SNAP !
- ✵ 3code.org组织和其他组织在中小学计算机教育中做的工作
- ✵ 4计算教育教师专业发展的方法和资料

CNCC CODE.ORG组织

- Code.org®是一个非营利组织，致力于扩大计算机科学在学校的应用，并增加年轻女性和其他代表性不足群体的学生的参与。我们的愿景是，每个学校的每个学生都有机会学习计算机科学，作为他们K-12核心教育的一部分。作为美国最大学区的K-12计算机科学课程的主要提供者，Code.org还创建了一年一度的 "一小时代码 "活动，该活动已经吸引了世界上所有学生的15%以上。
- Code.org得到了慷慨的捐助者的支持，包括微软、Facebook、亚马逊、Infosys基金会、谷歌和其他许多人。

CNCC CSFORALL

- CSforALL是对K-12计算机科学（CS）教育感兴趣的个人和组织的一个中心资源。将提供者、学校和地区、资助者和研究人员联系起来，致力于为美国的每个孩子提供高质量的CS教育。

- https://www.csforall.org/

CNCC TEALS项目

- 学校技术教育和扫盲（TEALS）是微软慈善机构的一个项目，在高中建立可持续的CS项目。专注于为因种族、性别或地理原因而被排除在CS学习之外的学生服务。TEALS通过将教师与行业志愿者和经过验证的课程配对，帮助他们学习教授CS。
- 自2009年成立以来，已有超过93,000名学生接受了CS教育。
- https://www.microsoft.com/en-us/teals

CNCC NCWIT

- 全国妇女和信息技术中心（NCWIT）是一个影响最深远的变革领导者网络，专注于通过纠正计算机领域的代表性不足来推动创新;
- NCWIT利用具有不同背景的专业人士的技能和经验来推进他们的使命，并进行可持续的变革。
- https://ncwit.org/

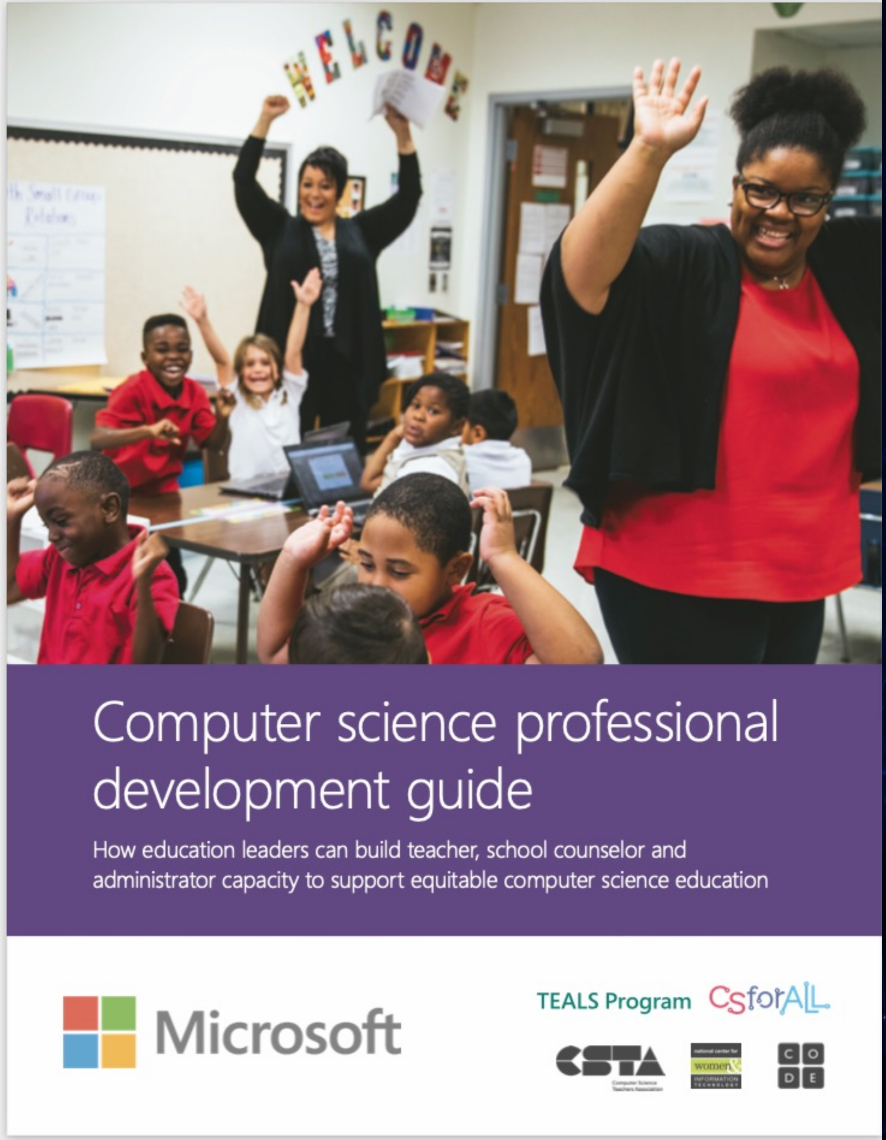
CNCC 计算教育教师专业发展的方法和资料

- 1美国中小学计算机科学教育的框架的核心概念和核心实践
- 2美国小学、初中和高中各个阶段的计算机课程
  - ☆ 课程路径
  - ☆ 比较Scratch, Alice, 和SNAP！
- 3code.org组织和其他组织在中小学计算机教育中做的工作
- 4计算教育教师专业发展的方法和资料

计算教育教师专业发展的方法和资料

教育领导者如何建设教师、学校辅导员和行政人员的能力以支持公平的计算机科学教育

第1步：确定项目的学生学习目标；
第2步：将教师与适合目标和优先事项的专业发展相匹配；
第3步：将教师与持续支持联系起来
第4步：争取和授权其他支持



Computer science professional development guide

How education leaders can build teacher, school counselor and administrator capacity to support equitable computer science education

Microsoft

TEALS Program CSforALL

CSTA

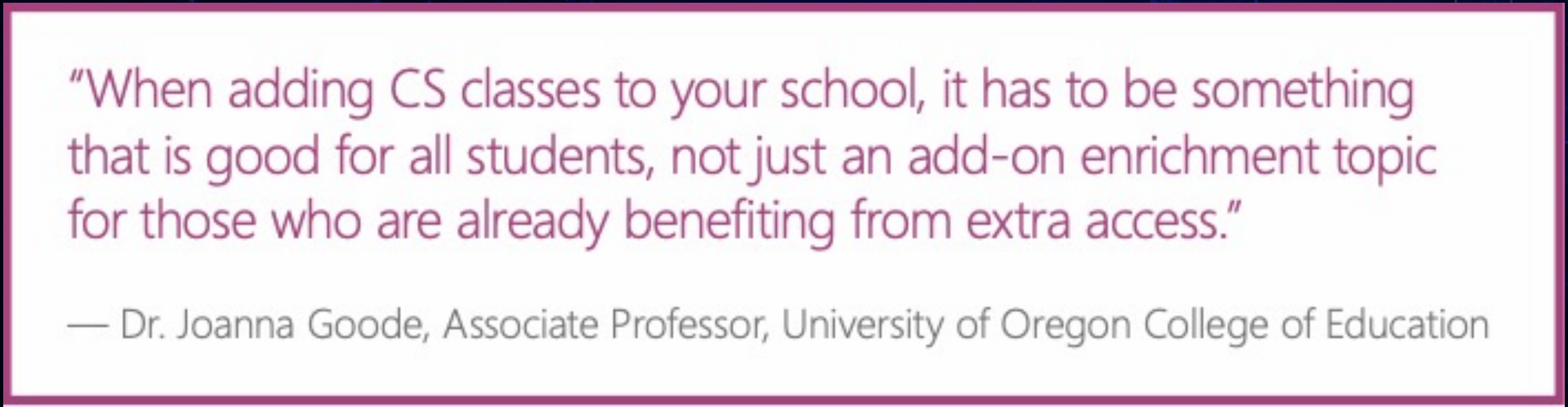
women & INFORMATION TECHNOLOGY

CODE

Computer Science Professional Development Guide
created by Microsoft, TEALS Program, CSforAll, NCWIT, CSTA, & Code.org in 2019

CNCC 1 确定项目的学生学习目标

- 建立一个CS领导小组，并与他们会面，讨论你的学校的项目、课程和整体理念；
- 了解与您所在州相关的CS学习和学习成效标准；
- 与教师合作，根据你的目标开发课程；
- 承诺对你的CS课程进行持续的评估和发展。

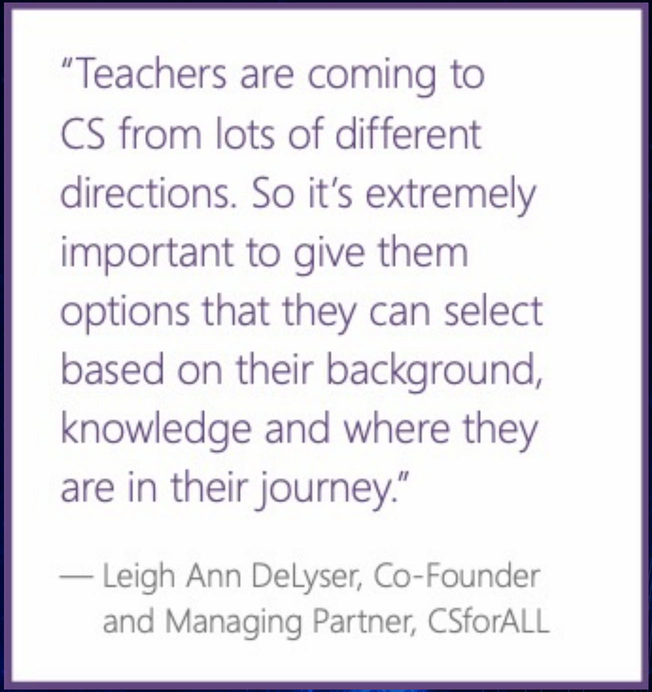


"When adding CS classes to your school, it has to be something that is good for all students, not just an add-on enrichment topic for those who are already benefiting from extra access."

— Dr. Joanna Goode, Associate Professor, University of Oregon College of Education

2 为教师匹配适合目标和优先事项的专业发展

- 初步的专业发展
  - 利用教师测试过的STEM和CS课程以及容易进入的学习机会；
  - 了解可以选择的许多专业发展供应商和产品；
  - 将教师与专注于创建和领导一个多样化和公平的课堂的机会和信息联系起来；
  - 通过让教师在Code.org上完成"一小时代码 "活动，向他们展示CS的有趣和创造性的一面。

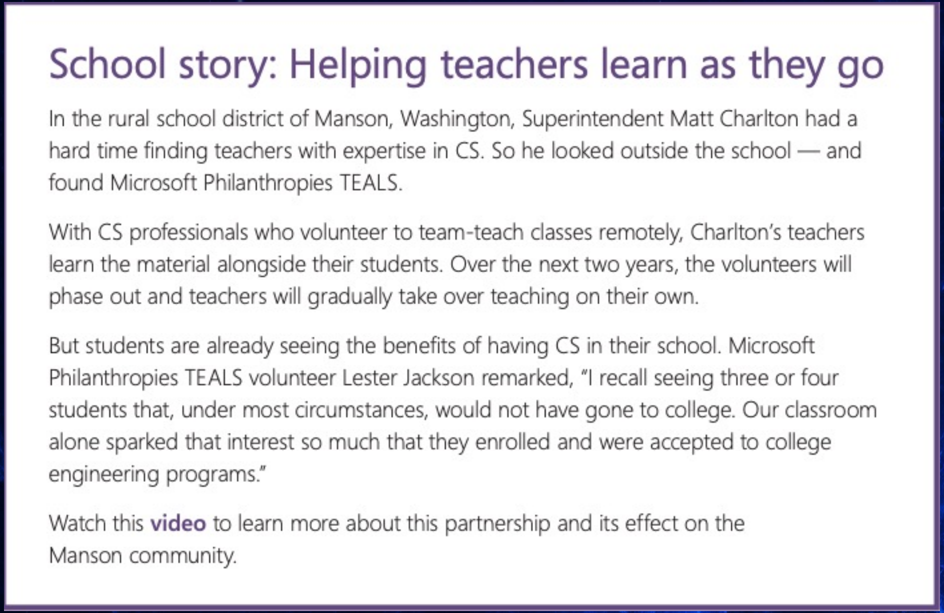


"Teachers are coming to CS from lots of different directions. So it's extremely important to give them options that they can select based on their background, knowledge and where they are in their journey."

— Leigh Ann DeLyser, Co-Founder and Managing Partner, CSforALL

2为教师匹配适合目标和优先事项的专业发展

- 持续的专业发展
  - 让教师参加专业发展会议；
  - 为你的教师研究认证机会；
  - 鼓励有经验的教师帮助新教师的初期专业发展；
  - 通过微软慈善基金会TEALS课堂强化模式，为教师提供来自行业志愿者的支持。



School story: Helping teachers learn as they go

In the rural school district of Manson, Washington, Superintendent Matt Charlton had a hard time finding teachers with expertise in CS. So he looked outside the school — and found Microsoft Philanthropies TEALS.

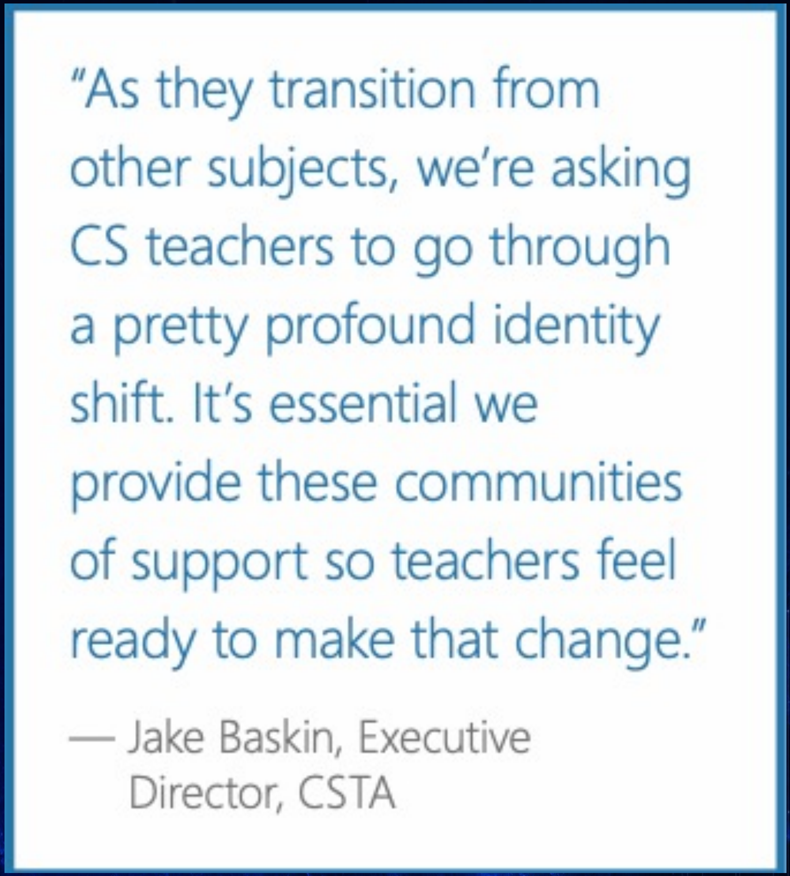
With CS professionals who volunteer to team-teach classes remotely, Charlton's teachers learn the material alongside their students. Over the next two years, the volunteers will phase out and teachers will gradually take over teaching on their own.

But students are already seeing the benefits of having CS in their school. Microsoft Philanthropies TEALS volunteer Lester Jackson remarked, "I recall seeing three or four students that, under most circumstances, would not have gone to college. Our classroom alone sparked that interest so much that they enrolled and were accepted to college engineering programs."

Watch this **video** to learn more about this partnership and its effect on the Manson community.

CNCC 3 将教师与持续的支持联系起来

- 将您所在地区的中小学教师聚集在一起，创建自己的教师工作小组；
- 鼓励和支持你的教师加入计算机科学教师协会（CSTA）；
- 探索国际教育技术协会（ISTE ）的个人和团体会员资格；
- 邀请教师加入 CS for All Teachers 的虚拟社区。

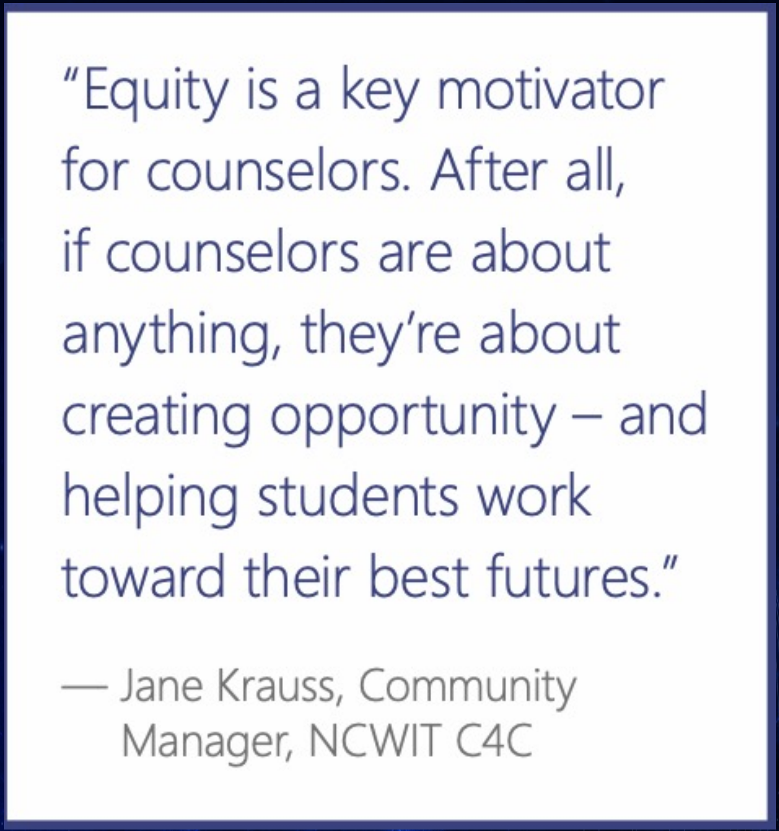


"As they transition from other subjects, we're asking CS teachers to go through a pretty profound identity shift. It's essential we provide these communities of support so teachers feel ready to make that change."

— Jake Baskin, Executive Director, CSTA

CNCC 4 争取和授权其他支持

- 辅导员的专业发展
  - ☆ 与你的辅导员会面，讨论他们在增加学生接受计算机教育和职业方面的作用。
  - ☆ 让辅导员在Code.org网站上完成 "一小时代码 "活动，向他们展示CS的有趣和创造性的一面。
  - ☆ 在你的地区举办计算机顾问(C4C)专业发展研讨会，不收取参与者任何费用。
  - ☆ 为辅导员提供工具，在CS课程中创造公平和机会。

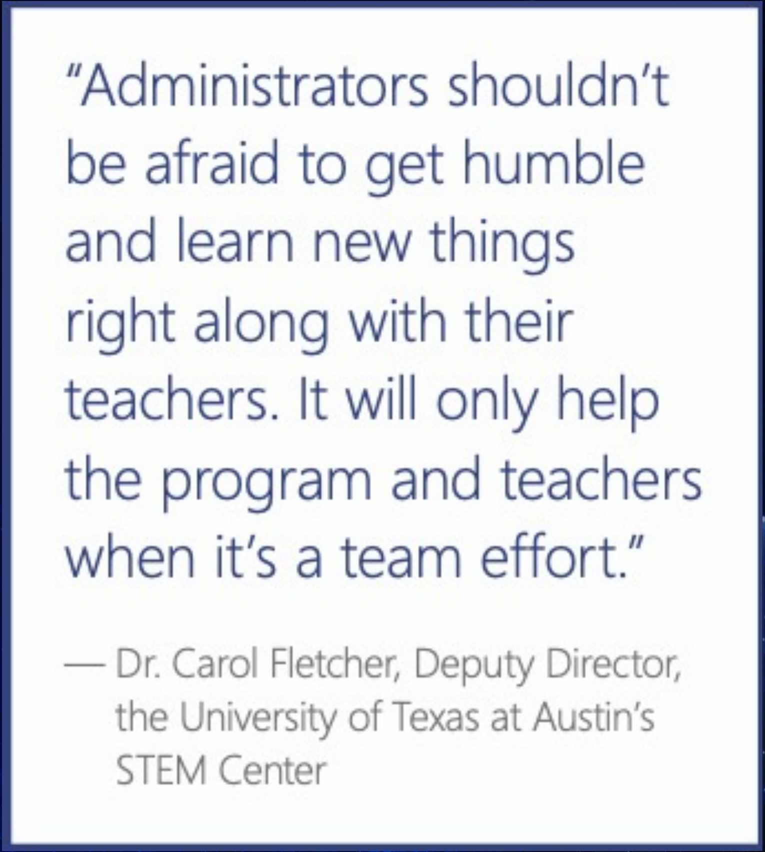


"Equity is a key motivator for counselors. After all, if counselors are about anything, they're about creating opportunity – and helping students work toward their best futures."

— Jane Krauss, Community Manager, NCWIT C4C

CNCC 4 争取和授权其他支持

- 管理者的专业发展
  - 在教师的专业发展过程中，参与一些相同的步骤，例如自己参加一些初步的和持续的专业发展；
  - 参加教师工作小组；
  - 参观其他学校的CS教室；
  - 学习如何使你的CS课程的入学率反映你的学校的整体人口状况。



"Administrators shouldn't be afraid to get humble and learn new things right along with their teachers. It will only help the program and teachers when it's a team effort."

— Dr. Carol Fletcher, Deputy Director, the University of Texas at Austin's STEM Center

CNCC 其他相关论文

- 《在大学通识教育中面向实践的科学和工程教育探索》 计算机教育 2019年第10期 《计算机教育》2019—2020 全国计算机教育优秀论文二等奖
- 《美国高中计算机教育对我国相应教育的启示》知识导刊 2021年第4期 2020年湖南省计算机教育年会征文二等奖

谢谢！