AP Computer Science Principals

Unit 1: Introduction to Programming

Purpose

Programming Focus: Introduction to Snap, Creating custom procedures, Abstraction

From the ***student's*** point of view, each of the programming labs is about what we hope will be an engaging project focus: Lab 1 is an interactive game suitable for use on a phone; Lab 2 is about conversation, between animated characters and between a character and the user; Lab 3 is about abstract art; and Lab 5 is a step toward storytelling animation through sprite interaction. But from the ***teacher's*** point of view, each has a coherent computer science goal: Lab 1 is a handholding-heavy, intellectually undemanding project to get students through the mechanics of using Snap*!* and to have a positive experience quickly; Lab 2 is mainly about functions (reporters) and composition of functions; Lab 3 is about traditional turtle graphics and abstraction; Lab 5 is about multithreaded programming and how sprites can be aware of each other.

Social Implications Focus: Digital Privacy

In Lab 4, students begin a yearlong inquiry into the social implications of computing by first examining the role of technology in their own privacy. They consider the innovations around us that collect data about us, the availability of information online, why privacy is good to protect, reasons for giving up privacy, and how to best protect their own online privacy. Review the teacher support on Facilitating Classroom Discussion.

For every discussion of social implications, try hard to get past "is this good or bad?" and instead focus on "what improvements to the design of the technology itself or to the social environment (such as laws and regulations) would keep the good parts without the bad parts?" This focus gives students an engineering perspective, and also a sense of agency; it's up to us—up to them, in the future—how technology is used.

Start class with five minutes of Computing in the News as often as possible (at least at the start of each week or lab, preferably daily). The goal is to quickly and regularly highlight how computing is constantly affecting our lives in new ways, without letting it take over the class period.

Big Ideas 1, 2, 4, 5, 7

Students are introduced to foundational concepts of **programming (Big Idea 5)** including loops, variables, and procedures, and apply these in small programming projects in which they are encouraged to explore creatively and embellish their programs to support the **creative (Big Idea 1)** development of interesting computational artifacts. As students learn to develop and evaluate **algorithms (Big Idea 4)** for implementation and execution on a computer, they see that complex programs can be simplified by using **abstraction (Big Idea 2)** to manage the complexity. Unit 1 includes

a Social Implications lab addressing online privacy, beginning a study of **global impact (Big Idea 7)** that is maintained throughout the course.

Computational Thinking Practices P3, P4, P5, P6

One way students develop their computational thinking skills is through programming and analyzing the programs of others. For example, students edit existing blocks and are asked to predict what they think will result from a given Snap*!* script. Making predictions requires students to make sense of the sequential nature of the code.

Students are introduced to one use of **abstracting (P3)** in packaging a collection of details in a form that more clearly expresses their meaning or purpose (such as creating a block that will draw polygons with specific characteristics (e.g., size, number of sides). They will expand their understanding of abstraction throughout the course.

The emphasis on Pair Programming in Unit 1 establishes the regular practice of **communicating (P5)** and **collaborating (P6)** to solve problems.

Students **analyze problems (P4)** about the broader world and **connect computing (P1)** to issues they can identify with—both the impacts of computing and the connections between people and computing. Students cultivate analytic thinking skills as they debate and sometimes write about important and complex social issues.

edX BJC Videos

There are a number of BJC videos from the edX version of the curriculum that are relevant to Unit 1. These are available on the edX BJC Videos page.

Student Labs

Lab 1: Click Alonzo Game

- 3-5 days (100-195 minutes)

Lab 2: Gossip and Greet

- 2-3 days (55-110 minutes)

Lab 3: Modern Art with Polygons

- 3-5 days (105-210 minutes) for required pages

Lab 4: Protecting Your Privacy

- 2-4 days (80-165 minutes)

Lab 5: Follow the Leader

Unit 2: Abstraction

Purpose

Programming Focus: Abstraction and Structure

In this unit students explore data structures (e.g., variables, lists, abstract data types) and program control structures (e.g., predicates, conditionals, higher order functions) that support abstraction. The general idea of abstraction is chunking a task into human-meaningful units and sweeping the details of those units under the rug. This makes a large project manageable because each chunk can be built and tested independently, and so that the composite of all the chunks is easier to read and debug. Of course, the programmer must do the under-the-rug work to build the abstractions, but once they work, those details can then be ignored. Students learn to test for special cases, to perform tasks and/or report information based on those cases. They also learn to create abstract data types and access and add list items in preparation for a deeper study of lists in Unit 3. These ideas are core to the AP CS Principles Abstraction standards and are core in both mathematical thinking and computer science.

Students also get experience finding and removing bugs from code, and learn how to incorporate old code that they've built into a new project that needs it.

Social Implications Focus: Media, Sharing, and Copyright

Students learn the meaning of copyrights and discuss their impact and relevance in an increasingly computationally mediated world.

Big Ideas 1, 2, 4, 5, 7

The major **programming (Big Idea 5)** focus of Unit 2 is on structure and **abstraction (Big Idea 2)**. Students learn to chunk details of **algorithms (Big Idea 4)** into meaningful, recognizable, potentially reusable parts, helping the top-level procedure show the structure of the program un-camouflaged by the details. This makes reading and debugging code easer and allows the smaller tool-procedures to be reused in other algorithms and to be refined without requiring revision to higher-level procedures. Students are offered a choice of programming projects designed to strengthen these ideas with **creative (Big Idea 1)** tasks. The unit includes a lab on copyright laws and issues addressing **global impact (Big Idea 7)**.

Computational Thinking Practices P1, P2, P3, P4, P5, P6

The focus on structure, **abstraction (P3)**, and debugging in Unit 2 supports students **analyzing problems and digital artifacts (P4)** as they determine where abstraction is needed and how to implement it and then critique their own code to improve it and resolve errors. The increased challenge of Unit 2's programs gives pair programmers good reason and plenty of practice to develop **communication (P5)** and **collaboration (P6)** skills because those skills are genuinely needed as

students create more complex **computational artifacts (P2)**. Discussions of social implications also help to develop communication and collaboration skills and create a context for broader discussion of more challenging social issues later in the year. In this unit, students **connect computing (P1)** to issues they may be able to identify with: creative rights in the digital era.

Student Labs

Lab 1: Improving Games by Using Variables

- 3-5 days (100-200 minutes)

Lab 2: Making Art by Using Data Structures

- 2-4 days (75-150 minutes) for required pages

Lab 3: Making Decisions by Using Predicates

- 3-5 days (105-210 minutes)

Lab 4: Dealing with Complexity

- 4-6 days (165-250 minutes) for required pages

Lab 5: Copyrights

Unit 3: Data Processing and Lists

Purpose

Programming Focus: Lists, Abstraction, Higher Order Functions

This unit focuses on lists, an *aggregate* data type for storing multiple items of any type, including numbers, strings, other lists, or even blocks. Just as functions can take numbers and strings as inputs, they can also take lists as input, or produce lists as output. A list is an ordered, numbered sequence of items. Similar data types in other programming languages may be called "arrays," "sequences," or "vectors."

This unit also includes the use of several powerful list-processing functions, including the **higher-order functions** `map`, `keep`, and `combine`. These functions are called "higher order" because, along with other data, they take functions as inputs. For example, often a programmer wants to compute some function of each item in a list. Instead of writing separate procedures such as "take the first letter of each item," "add 3 to each item," etc., the `map` function generalizes the "... of each item" part, and takes another function as input to specify the "first letter of" or "add 3 to" part.

One of several design features that distinguishes BJC from other CS Principles curricula is that we emphasize *functional* programming. One virtue of the higher order list functions is that they generate new lists to report, rather than mutate existing lists. This is in contrast with the imperative, looping, mutation-based programming that is more common, but more error-prone, in dealing with sequential data. No attention (for now) needs to be placed on the idea of procedures as data; the Snap*!* visual representation of higher order functions makes the use of a function (rather than the value it reports) as input apparent at a glance. The grey ring that represents a procedure as data is already in the higher order function block, and the user of the higher order function doesn't have to do anything else to make the function itself, rather than a value it reports, be taken as the input. Near the end of the course, students build these higher order functions for themselves.

In situations in which imperative programming is needed, we still use a higher order procedure, `for each item`, a C-shaped block that takes a list and a script to be run for each item in the list. This avoids the need for index variables.

Social Implications Focus: Robots and AI

Students are introduced to some ideas and issues in artificial intelligence and discuss the challenges and consequences of this new technology.

Big Ideas 2, 3, 4, 5, 7

As students work with lists to manage **data and information (Big Idea 3)**, they use **abstraction (Big Idea 2)**, such as abstract data types, to manage the complexity of datasets. They also create **algorithms (Big Idea 4)** using higher-order functions and implement these algorithms in various **programs (Big Idea 5)** that use lists and list functions.

In the Social Implications Labs, students continue their study of the **global impact (Big Idea 7)** of computing, now by focusing on the impact of machine learning and robotics.

Computational Thinking Practices P1, P2, P3, P6

Lists are an important data structure, and learning to think of lists as a single object (rather than many objects) is a way students practice **abstracting (P3)**. We advance the idea of abstract data types by creating more complex, nested ADTs. Students continue to **collaborate (P6)** through pair programming.

Several labs are built around a small "app" project, such as a contact list app or augmenting their Tic Tac Toe game, a **computational artifact (P2)** created by the student. This **connects computing (P1)** with the apps and games students are already familiar with.

Student Labs

Lab 1: Contact List

- 2-4 days (75-140 minutes)

Lab 2: Tic-Tac-Toe

- no required pages

Lab 3: Robots and Artificial Intelligence

- 4-8 days (150-330 minutes)

Lab 4: Building Data Visualization Tools

- no required pages

Lab 5: Big Data

- 2-5 days (75-190 minutes)

Assessment: Tic Tac Toe Analysis of Abstractions

Students complete and refine their work on the Tic-Tac-Toe project, describe the purpose of the program (in 100 words or less), and write a description of the ***abstractions*** they used (in preparation for the AP Create Task). This Unit 3 Asssessment file includes student-facing instructions and a list of the abstractions students have used in developing this project.

Unit 4: How the Internet Works

Purpose

Programming Focus: Network Protocols, Cybersecurity

Unit 4 addresses the structure of the Internet and the various protocols on which it runs and the implications of this technology to society. From the AP CSP Framework: "The Internet pervades modern computing. The Internet and the systems built on it have had a profound impact on society. Computer networks support communication and collaboration. The principles of systems and networks that helped enable the Internet are also critical in the implementation of computational solutions. Students in this course

gain insight into how the Internet operates, study characteristics of the Internet and systems built on it, and analyze important concerns such as cybersecurity." Unit 4 addresses these ideas.

Social Implications Focus: Online interactions and Community; Impact of Computers on Work

Lab 4: Community and Online Interactions explores the ways in which innovations in computing have impacted how we communicate as human beings and how our online interactions are shaping our ability to build a community. Cyberbullying and free speech are some of the issues discussed in depth.

Lab 5: Computers and Work examines how progress in computing technology is impacting work and considers the issues and challenges surrounding the 21-century work environments.

There is a lot of reading in this unit. Mix it up to keep students engaged. Encourage students to take turns reading in small groups or as a class, or consider assigning different teams to the various pages of the lab and have each team present their section to the class. Choose a strategy that will work for your group of students.

Big Ideas 2, 6

The primary focus of this unit is the **Internet (Big Idea 6)**. Students learn about the power of the fundamental **abstractions (Big Idea 2)** of the Internet including the two most essential protocols: IP, which builds one Internet out of a vast collection of local networks; and TCP, which allows the Internet to function reliably despite the lack of reliability of the physical infrastructure. Students also learn how the hierarchies of domain names and IP addresses make scalability possible.

Computational Thinking Practices P1, P3, P5, P6

Students learn to identify and describe the **abstractions (P3)** of the Internet, continue to **collaborate (P6)** as they pair program, and **communicate (P5)** as they **connect computing (P1)** to society in the Social Implications labs.

edX BJC Videos

There are a number of BJC videos from the edX version of the curriculum that are relevant to Unit 4. These are available on the edX BJC Videos page.

Student Labs

Lab 1: Reliable Communication

- 2-3 days (60-120 minutes) for required pages

Lab 2: Communication Protocols

- 2-4 days (60-170 minutes)

Lab 3: Cybersecurity

- 4-6 days (180-240 minutes) for required pages

Lab 4: Community and Online Interactions

- 3-5 days (120-220 minutes)

Lab 5: Computers and Work

- 2-3 days (80-120 minutes)

Assessment: A Collaborative Class Project

The Internet is vast and ever-evolving, so it is not an easy task to cover it in a single unit. A possible approach is a divide-and-conquer Collaborative Class Project. Each student, perhaps with a partner, will prepare a 5-minute presentation on a particular topic of interest, drawn from the AP CSP standards.

Unit 5: Algorithms and Data

Purpose

Programming Focus: This unit focuses on several types of analysis: analysis of problems to generate algorithms for their solution; analysis of the algorithms, especially of the time it takes to execute them, to optimize them; analysis of phenomena to generate models and simulations that give insight and help one generate and test hypotheses; and analysis of data, especially including visualization.

Students have been generating algorithms to solve problems from the start of this course, but have not yet focused on analyzing them for efficiency. For small enough computational problems, such analysis isn't needed. But modeling complex phenomena and handling large data sets requires understanding that there are sometimes alternative algorithms that reduce the impact of the size of a model on the time it takes to execute. In-depth coverage of this broad domain (computational complexity, data analysis, modeling and simulation) is beyond the scope of an introductory course, but this unit's projects in each of these areas will give students a good first-approximation understanding of these issues.

Social Implications Focus: Computing in War

Students learn how computing innovations impact modern warfare and reflect upon the effects of this impact on society.

Big Ideas 2, 3, 4, 5

The big focus of Unit 5 is on analyzing **algorithms (Big Idea 4)** and **data and information (Big Idea 3).** Students explore various ways to process, create, analyze, and visualize data through their experiments and simulations relying and building on their experience with **programming (Big Idea 5)** and **abstraction (Big Idea 2)**.

Computational Thinking Practices P1, P2, P4, P5, P6

Students **create** several **computational artifacts (P2)** including an extension of the number guessing game from Unit 2 and a coin flipping simulator. Students **analyze problems** (e.g., list searching algorithms and data scaling) to create suitable algorithms for solving them, and they create a timer program to help them **analyze programs (P4)** for their computational efficiency. Students continue to **connect computing (P1)** to the human experience through their reading and discussions about copyrights and to **communicate (P5)** and **collaborate (P6)** through pair programming.

edX BJC Videos

There are a number of BJC videos from the edX version of the curriculum that are relevant to Unit 5. These are available on the edX BJC Videos page.

Student Labs

Lab 1: Search Algorithms

- 3-6 days (120-240 minutes) for required pages

Lab 2: Models and Simulations

- 2-4 days (70-140 minutes) for required pages

Lab 3: Timing Experiments

- 3-6 days (120-240 minutes)

Lab 4: Unsolvable and Undecidable Problems

- 2-4 days (70-140 minutes)

Lab 5: Computing in War

- 3-4 days (105-180 minutes)

Optional Project: Tic-Tac-Toe

Guidance for the AP Performance Tasks

Links

- AP CSP: The Exam - official College Board site with exam information, scoring webinar recording, and sample performance tasks
- BJC Student Performance Tasks Page
  - AP CSP Explore Task
  - AP CSP Create Task

Pacing

- Explore: Impact of Computing Innovations—Teachers must provide **8 classroom hours** to complete the Explore Task.
- Create: Applications from Ideas—Teachers must provide **12 classroom hours** to complete the Create Task.

Tips

- The BJC Performance Task pages are designed to be a *support* to students as they complete these College Board tasks and should be treated as a supplement to the official College Board materials. The College Board website and documentation should be consulted regularly and considered the final word on the requirements of the Tasks. The BJC pages are not intended to be a complete resource and may not contain the most up to date information. They are provided to help clarify requirements that have been confusing to students.
- For the Create Task, consider drawing as big a crowd as possible for the students to show off their work (other students, teachers, parents, and administrators).

Example Student Snap! Project Videos

- Texas Holdem
- Minesweeper
- Inception Tic-Tac-Toe

Related edX BJC Videos

No YouTube access at your school?
Try these Alternate Links.

- Creativity: Introduction

- Creativity: Computational Artifacts
- Creativity: Collaboration, Analysis, Power! Part 1
- Creativity: Collaboration, Analysis, Power! Part 2
- Human Computer Interaction Part 1
- Human Computer Interaction Part 2
- Human Computer Interaction Part 3
- Human Computer Interaction Part 4
- Human Computer Interaction Part 5

Correlation with AP CS Principles Framework

Enduring Understandings for Explore Task:

- **EU 7.5** An investigative process is aided by effective organization and selection of resources. Appropriate technologies and tools facilitate the accessing of information and enable the ability to evaluate the credibility of sources.

Learning Objectives for Explore Task:

- **LO 1.2.1** Create a computational artifact for creative expression. [P2]
- **LO 1.2.2** Create a computational artifact using computing tools and techniques to solve a problem. [P2]
- **LO 1.2.5** Analyze the correctness, usability, functionality, and suitability of computational artifacts. [P4]
- **LO 3.3.1** Analyze how data representation, storage, security, and transmission of data involve computational manipulation of information. [P4]
- **LO 7.1.1** Explain how computing innovations affect communication, interaction, and cognition. [P4]
- **LO 7.3.1** Analyze the beneficial and harmful effects of computing. [P4]
- **LO 7.4.1** Explain the connections between computing and real-world contexts, including economic, social, and cultural contexts. [P1]
- **LO 7.5.1** Access, manage, and attribute information using effective strategies. [P1]
- **LO 7.5.2** Evaluate online and print sources for appropriateness and credibility. [P5]

Essential Knowledge for Explore Task:

- **EK 1.2.5A** The context in which an artifact is used determines the correctness, usability, functionality, and suitability of the artifact.
- **EK 1.2.5B** A computational artifact may have weaknesses, mistakes, or errors depending on the type of artifact.
- **EK 1.2.5C** The functionality of a computational artifact may be related to how it is used or perceived.
- **EK 1.2.5D** The suitability (or appropriateness) of a computational artifact may be related to how it is used or perceived.

- **EK 7.5.1B** Advanced search tools, Boolean logic, and key words can refine the search focus and/or limit search results based on a variety of factors (e.g., peer-review status, type of publication.
- **EK 7.5.1C** Plagiarism is a serious offense that occurs when a person presents another's ideas or words as his or her own. Plagiarism may be avoided by accurately acknowledging sources.
- **EK 7.5.2A** Determining the credibility of a source requires considering and evaluating the reputation and credentials of the author(s), publisher(s), site owner(s), and/or sponsor(s).
- **EK 7.5.2B** Information from a source is considered relevant when it supports an appropriate claim or the purpose of the investigation.

Learning Objectives for Create Task:

- **LO 2.2.1** Develop an abstraction when writing a program or creating other computational artifacts. [P2]
- **LO 4.1.1** Develop an algorithm for implementation in a program. [P2]
- **LO 4.1.2** Express an algorithm in a language. [P5]
- **LO 5.1.1** Develop a program for creative expression, to satisfy personal curiosity, or to create new knowledge. [P2]
- **LO 5.1.2** Develop a correct program to solve problems. [P2]
- **LO 5.2.1** Explain how programs implement algorithms. [P3]
- **LO 5.3.1** Use abstraction to manage complexity in programs. [P3]
- **LO 5.4.1** Evaluate the correctness of a program. [P4]
- **LO 5.5.1** Employ appropriate mathematical and logical concepts in programming. [P1]