AP CS Principles: Beauty and Joy of Computing

Course Overview:

BJC covers the entire 2020 CS Principles Framework and addresses the five Big Ideas with a primary emphasis on programming and abstraction (Big Idea 3: Algorithms and Programming). As much as possible, BJC uses programming as the vehicle to tell other parts of the story; for example, by presenting data (Big Idea 2: Data) not through commercial database software but by writing programming projects that manipulate data sets as lists.

The secondary emphasis of BJC is on the social implications of computers (Big Idea 5: Impact of Computing). Social topics are included in every unit, and students are encouraged to think critically about each application of technology. Purpose-driven program development (Big Idea 1: Computational Solution Design) is addressed throughout the units with special focus in the Practice Create Task. And there is particular attention to the Internet (Big Idea 4: Computing Systems and Networks) in Unit 4.

Snap*!*, the programming language used in BJC, was developed specifically for this curriculum. Its visual, drag-and-drop design is based on that of Scratch, so that it is accessible to a wide audience and not intimidating, but the language, itself, is extended with the abstraction mechanisms needed for serious computer science: first class procedures for control abstraction and first class lists for data abstraction. These capabilities are embodied in carefully chosen visual metaphors so that ideas traditionally considered difficult can be understood and enjoyed by beginners.

Course Resources (all free):

- Beauty and Joy of Computing curriculum (bjc.edc.org)
- Beauty and Joy of Computing Teacher Guide (bjc.edc.org/teacher; visit bjc.link/guideaccess for initial access)
- Blown to Bits. Abelson, H., Ledeen, K., & Lewis, H. 2008, Addison-Wesley. (bitsbook.com/thebook; scroll down for free PDF)
- Snap*!* programming language (snap.berkeley.edu/run)

Course Outline:

Unit 1: Introduction to Programming

Students develop an interactive game they can install on their phones, generate a conversation between animated characters, create abstract art, and explore storytelling animation through sprite interaction (**AAP**); and in doing so, they learn to use pair programming and to create program documentation. (**CRD**) Students also investigate legal and ethical issues that arise in computing—especially with regard to data collection and privacy (**IOC**).

Unit 2: Abstraction

Students implement an algorithm for a guessing game using local and global variables; use abstract data types and list traversal to build a quiz app; create predicates to filter lists in order to solve a crossword puzzle; and use the modulus function and a higher order function to code mathematical functions (**AAP**). Students also investigate the history, purpose, laws, evolution, and enforcement of copyright (**IOC**).

Unit 3: Data Structures

Students explore complexity in a variety of contexts (maze navigation, fractal art, tic-tac-toe); use nested abstract data types and data I/O to develop a contact list app; and consider the beneficial and harmful impacts of robots and AI (**CRD**, **AAP**, **IOC**).

Practice AP Create Task

Students create a project of their own choosing as practice for the AP Create Task. They select and use a development process, plan and code their program, test their program for errors, write about their development process, and acknowledge any code developed by other people (**CRD**).

Unit 4: How the Internet Works

Students learn about how the Internet works, the benefits and vulnerabilities of fault-tolerant systems; cybersecurity practices such as public key encryption and individual level practices and software to keep data safe; digital data representation including binary representation; compression algorithms (**CSN**, **IOC**, **DAT**). They also consider the impact of the Internet on human communication and the workplace (**CRD**, **IOC**).

Unit 5: Algorithms and Simulations

Students learn about program efficiency through exploration of the binary and linear search algorithms; learn about sequential, parallel, and distributed computing and determine the contexts in which each are most useful; consider the contexts in which simulation is useful and implement a simple simulation; use Snap*!* data tools to generate knowledge from data (**AAP**, **CSN**, **DAT**).

AP Create Task

Students complete the *AP Create Task* (12 hours in class).

Note: Units 1-5, the Practice Create Task, and the Create Task cover the CSP curriculum framework. Units 6-8 contain additional material that's important to BJC including the abstraction hierarchy of how computers work, recursion and functional programming.

Unit 6: How Computers Work

Building on their understanding of abstraction and the way computers store data, students learn about the computer system abstraction hierarchy, with application software on top and transistors at the bottom.

Unit 7: Fractals and Recursion

Students deepen their experience with recursion and functional programming through drawing projects that use recursive commands, mainly fractals.

Unit 8: Recursive Functions

Students extend their understanding of abstraction and recursion through exploration of recursive functions: sorting lists (both selection sort and partition sort), Pascal's triangle, converting numbers to and from binary, finding the subsets of a set, and building several higher order functions from scratch.

Example Activities Aligned to AP Big Ideas:

Big Idea 1: Creative Development

In the Practice Create Task, students plan the development process (**CRD-2.E**) of a project of their own choosing, plan their program's behavior and how they will meet their own specifications (**CRD-2.F**), acknowledge code written by other people (**CRD-2.H**), test their code by identifying inputs and expected outputs (**CRD-2.J**), and identify and fix any problems with their code (**CRD-2.I**).

Big Idea 2: Data

Unit 5 Lab 3: Turning Data into Information begins with an introduction to data analysis in which students analyze and describe their findings about global health data (**DAT-2.A**) and describe what they learn through the interactive interface **(DAT-2.E**). The majority of the lab consists of a data processing project in which students import a database and answer questions about their dataset (**DAT-2.D**). Students also describe possible challenges of collecting data (**DAT-2.C**) and what information can be extracted from metadata (**DAT-2.B**).

Big Idea 3: Algorithms and Programming

In Unit 2 Lab 1: Games, students build a number-guessing game in which the player tries to guess the computer's secret number (**AAP-2.B**). They use a local variable to store the secret number and a global variable to store the player's score (**AAP-1.A**) and use random numbers (**AAP-3.E**) to model language and to generate the secret number. In doing so, they learn how to assign a value to a variable and how to determine the result of sequential variable assignments (**AAP-1.B**), and they use lists to store the computer player's character costume options (**AAP-2.N**).

Big Idea 4: Computing Systems and Networks

Unit 4 Lab 1: Computer Networks introduces the structure and protocols of the Internet. Students learn how the Internet is different from the World Wide Web (**CSN-1.D**); how computing devices are connected in a network (**CSN-1.A**); the benefits of, features of, and need for fault tolerance (**CSN-1.E**); and how data are sent through the Internet via packets (**CSN-1.C**). And they combine these ideas into a short paper on how the Internet works (**CSN-1.B**).

Big Idea 5: Impact of Computing

In Unit 4 Lab 3: Community and Online Interactions, students examine ways in which computing affects our ability to build community, including the digital divide (**IOC-1.C**) and the impact of crowdsourcing on scientific research and fundraising (**IOC-1.E**); and they write about the purpose and unintended consequences of a computing innovation of their choosing (**IOC-1.B**).

Example Activities Aligned to AP Computational Thinking Practices:

Practice 1: Computational Solution Design

In the Practice Create Task, students prepare for the official Create Task by designing a program of their own choosing. Students carefully choose a program concept that will meet the guidelines of the Create Task and then write about their decision-making process as well as the program's purpose (**CPT 1**).

Practice 2: Algorithms and Program Development

In Unit 3 Lab 1 Page 1: Robot in a Maze, students use sequencing, selection, and repetition to plan an algorithm for escaping from a maze without actually programming the algorithm (**CPT 2**).

Practice 3: Abstraction in Program Development

Unit 3 emphasizes the study of abstraction as the means to control complexity. In Lab 1 Page 4: Brick Wall, students use abstraction to create a somewhat complex program built on relatively simple custom blocks that specialize on specific tasks (e.g., building one brick, building alternating rows of bricks, building a brick wall). Then in Lab 2: Contact List, they build on their use of abstract data types in Unit 2 to develop a constructor that creates a contact to be added to a contact list and several selectors for accessing the name, address, or phone number for any given contact. Students then extend their use of abstract data types to create a nested abstract data type to store and retrieve components of the birthdate of each contact (**CPT 3**).

Practice 4: Code Analysis

Students begin evaluating and testing their code in Unit 1. In Lab 2: Gossip, they determine the result of running code that manipulates strings and code that generates random values, and they make prescribed changes to their own code and describe the resulting change in program behavior as well as the cause. In Lab 3: Modern Art with Polygons, they learn the purpose and practice of program documentation and learn to write, call, and predict the result of calling procedures (**CPT 4**).

Practice 5: Computing Innovations

In Unit 1 Lab 4: Protecting Your Privacy, students consider the information that is available online about them, discuss why privacy is good to protect, consider reasons for giving up privacy, and discuss the benefits and risks of various privacy-impacting computing innovations (**CPT 5**).

Practice 6: Responsible Computing

In Unit 4 Lab 2: Cybersecurity, students learn about various cybersecurity risks on the Internet, what they can do to protect themselves online, and the basic concepts of cryptography (**CPT 6**).

Opportunities to Investigate Computing Innovations:

Prompt A: Beneficial and Harmful Effects

In Unit 4 Lab 3 Page 6: Benefits of Computing, Exercise 3, students select a computing innovation and write about its original purpose and unintended consequences, creating a two-by-two chart of consequences with axes good/bad, intended/unintended.

Prompt B: Identifying and Explaining the Use of Data

In Unit 5 Lab 3 Page 4: Analyzing Data, Exercise 5, student select a computing innovation that uses a lot of data, and describe the kinds of data it uses, the origins of those data, and how the application transforms the data to extract information.

Prompt C: Privacy, Security and Storage Concerns

In Unit 1 Lab 4 Page 4: Innovations and Privacy, Exercise 6, students select a computing innovation and describe the privacy concerns that it raises.