CAREER & TECHNICAL EDUCATION

HIGH SCHOOL ADVANCED COMPUTER PROGRAMMING

ACADEMIC & PROFESSIONAL SKILLS STANDARDS

CATALINA FOOTHILLS SCHOOL DISTRICT
Approved by the Governing Board
April 27, 2021

STANDARDS FOR COMPUTER PROGRAMMING

ADVANCED COMPUTER PROGRAMMING

Advanced Computer Programming is a college-level course that introduces students to applied software design and engineering principles (alongside the algorithms which drive them) through hands-on, project-oriented coursework. Students will design, develop, and test their own industry project (e.g., an app, video game, webpage, or other software project) using industry standard programming languages, skills and software development expert-standards. Projects will be developed using industry standard programming languages from which students will be introduced to industry-level programming skills and software development expert-standards. This course will be fast-paced and require a significant level of effort, initiative, independent study/practice, and self-discipline, as well as a strong background in Algebra and/or Discrete Mathematics. Students will be prepared to attain advanced industry certifications in the programming language(s) used.

CODING STYLE AND RESOURCES

- ACP.1.1 Apply appropriate code indentation and comment practices when writing software applications.
- ACP.1.2 Utilize a code repository to manage programming projects and view/retrieve code available thereon for public use (i.e., via MIT CC License).
    - Consistently and appropriately indent lines of code to make it easier to understand and detect bugs.
    - Use a code hosting platform to download/upload code and collaborate on a shared coding project; through using commands such as pull, push, commit.

CONSTRUCTS (FROM PRIMITIVE VALUES TO OBJECTS)

- ACP.2.1 Describe an expression, statement, and block of code in Java, including their relative scope.
- ACP.2.2 Explain the fundamentals of encapsulation in Java, as well as the technicals via implementation in code.
- ACP.2.3 Distinguish between an object class and static class, and apply both appropriately within code projects.
    - Explain how executable code in many languages (including Java) are composed of blocks, statements and/or expressions.
    - Describe how the state and behavior[s] of something in the world (i.e., described in a design document) correlate to the variables and methods of its object representation in code.
    - Describe the concept of encapsulation and how the state and behaviors of an object class implement it towards representing instances of that class.
    - Explain an object class, a static class, and implement each in code.
    - Identify and demonstrate where each kind of class is [best] utilized within software code.

APPLICATIONS OF LOGICAL AND CONTROL-FLOW STATEMENTS

- ACP.3.1 Utilize logical operators and conditional (i.e., decision-making) statements in a variety of use-cases and scenarios.
- ACP.3.2 Utilize iterative (i.e., looping) statements in a variety of use-cases and scenarios.
    - Suggest the logical operator[s] and/or conditional statement[s] to use towards implementing a correct solution to a logic-based computational task; while being able to explain the suggestion offered.
    - Choose between using an {if, else-if, else} statement versus a switch statement for a given scenario, and explain why the choice was made.
    - Identify the fundamental difference between each type of loop {while, for, do while, foreach}, suggest one to use for a given scenario, and explain the choice made.

JAVA BUILT-IN OBJECT TYPES

- ACP.4.1 Describe and implement a variety of methods from the Java String and Stringbuilder classes, as well as [print] format methods.
- ACP.4.2 Describe and implement a variety of methods from the Java array class through the utilization of coding drills.
- ACP.4.3 Describe and implement a variety of methods from classes of the Java List and Collections interface (i.e., ArrayList, LinkedList, Stack, Queue, HashMap/Set).
- ACP.4.4 Describe and implement a variety of methods from utility classes of Java (i.e., Time, Date, Period, Math).
- ACP.4.5 Describe and implement Java lambda expressions.
    - Perform string operations such as: sorting a collection of strings in lexicographic order, parsing, formatting, and reading from / writing to text files.
    - Implement several integer sorting algorithms (i.e., Selection, Insertion, and Merge Sort) using a set of values within an input integer array.
    - Describe the Abstract Data Type (ADT) and operations thereof for {stack, queue, hash table, linked list}.
    - Implement simple stacks and queues with both arrays and linked lists (including ADT operations e.g., {push, pop, peek, append, prepend, enqueue, dequeue, etc.}).
    - Explain how Java lambda statements work, suggest use-cases, and implement them in coding projects which require (or hint to using) them.

APPLICATIONS OF OBJECT-ORIENTED PROGRAMMING (OOP)

- ACP.5.1 Utilize Java inheritance and polymorphism towards implementing well-designed OO executable code.
- ACP.5.2 Utilize Java interfaces and abstract classes towards implementing well-designed OO executable code.
- ACP.5.3 Explain the fundamentals and benefits of composition and its implementation in Java.
- ACP.5.4 Explain the fundamentals of Java generics and their applications in software design/development.
    - Explain the fundamentals of class inheritance, offering pros/cons of using it as opposed to other designs (i.e., entity-component), and discuss possible use cases.
    - Explain the fundamentals of interfaces, offering pros/cons of using it as opposed to other designs (i.e., entity-component), and discuss possible use cases.
    - Explain the fundamentals of abstract classes, offering pros/cons of using it as opposed to other designs (i.e., entity-component), and discuss possible use cases.
    - Identify and describe possible uses of composition and polymorphism in examples such as video games, social media apps, etc.
    - Describe the main ideas of Java generics, identify where they are used in Java's built-in classes, and discuss several possible use-cases for them.

JAVA EXCEPTION HANDLING AND DEBUGGING

- ACP.6.1 Utilize Java constructs (e.g., try/catch blocks) to handle exceptions, alongside other techniques to prevent several common exception scenarios from ever occurring (e.g., input checking, null value checking, etc.).
- ACP.6.2 Utilize Java unit testing as a technique to implement correctly working code via testing for correctness as it's being implemented, as well as for debug purposes.
- ACP.6.3 Utilize debugger utilities provided in IDE's such as Eclipse or Netbeans to debug code.
    - Implement {try, catch, finally} statements in code where appropriate to handle various semantic or otherwise errors; and identify where such should go if given a method's code and adequate documentation of inputs/outputs/purpose/etc. (i.e., diagnose).

- Set up a unit test system/utility (in Eclipse WLOG), write substantive test cases for it, and actively use it when writing code for assignments and other projects.
- Utilize a debugger's breakpoint system and profiler[s] to 'step through' executing code (to find where it had problems) and analyze memory/disk performance. (e.g., viruses, phishing, suspicious email, social engineering, spoofing, identity theft, and spamming).

ADVANCED TOPICS IN SOFTWARE DESIGN

- ACP.7.1 Explain the origin, purpose, and advantages of [software] design patterns.
- ACP.7.2 Describe several of the main design patterns (i.e., in terms of the task[s] that a design pattern performs and/or a problem that they solve), and demonstrate their advantages through code projects.
- ACP.7.3 Describe several Data Structures, and demonstrate their basic state and methods through code projects.
    - Describe the Command design pattern, including purpose, advantages, and use-cases.
    - Describe the Composite design pattern, including purpose, advantages, and use-cases.
    - Describe the Factory design pattern, including purpose, advantages, and use-cases.
    - Describe the Observer design pattern, including purpose, advantages, and use-cases.
    - Describe the Binary Search Tree data structure, including purpose, advantages, and use-cases.
    - Describe the Quadtree data structure, including purpose, advantages, and use-cases.
    - Describe the Skiplist data structure, including purpose, advantages, and use-cases.
    - Describe the Voronoi Diagram data structure, including purpose, advantages, and use-cases.

PROFESSIONAL SKILLS: PROFESSIONALISM & ORGANIZATIONAL CULTURE

- ACP.8.1 Represent the school [organization] in a positive manner, demonstrating the school's [or organization's] mission and core values.
    - Communicate the mission and core values of the school [or organization].
    - Perform my work with a positive attitude.
- ACP.8.2 Demonstrate professionalism in the workplace (being on time, proper dress, courteousness).
    - Follow protocol(s) related to behavior, appearance, and other expectations.
    - Explain the importance of "dress for success."
- ACP.8.3 Demonstrate respect for personal and professional boundaries.
    - Distinguish between personal and work-related matters.
- ACP.8.4 Interact respectfully with others (cross-cultural, intergenerational, individuals with disabilities); act with integrity.
    - Address challenges with sensitivity.
- ACP.8.5 Produce high-quality work that reflects professional pride and organizational values, and contributes to organizational success.
    - Create work products in a timely manner that are high quality and positively represent the organization.
- ACP.8.6 Take initiative to develop skills and improve work performance.
    - Identify and apply strategies to improve my performance.

PROFESSIONAL SKILLS: COMPLEX COMMUNICATION (TRADITIONAL AND DIGITAL)

- ACP.9.1 Communicate effectively in a diverse work environment (i.e., style, format, and medium appropriate to audience/culture/generation, purpose and context; accuracy; use of appropriate technical/industry language; to resolve

conflicts; address intergenerational differences/challenges; persuade others).

- ACP.9.2 Use documentation (e.g., itineraries and schedules) to plan and meet client needs.
    - Use appropriate verbal and nonverbal modes of communication.
    - Proof and edit all communications based on [organizational] standards
    - Address communications in a style that is appropriate to the audience and situation.
    - Respond in a timely manner to communications.
    - Verify the accuracy of information and authority of sources.
- ACP.9.3 Use appropriate technologies and social media to enhance or clarify communication.
    - Use professional etiquette and follow applicable laws and regulations for web-, email-, and social media-based communications.
- ACP.9.4 Use a variety of interpersonal skills, including tone of voice and appropriate physical gestures (e.g., eye contact, facing the speaker, active listening) during conversations and discussions to build positive rapport with others.
    - Demonstrate appropriate active listening skills.
- ACP.9.5 Pose and respond to questions, building upon others' ideas in order to enhance the discussion; clarify, verify, or challenge ideas and conclusions with diplomacy.
    - Ask questions to obtain accurate information.

PROFESSIONAL SKILLS: INITIATIVE AND SELF-DIRECTION

- ACP.10.1 Apply the skills and mindset of self-direction/self-regulation to accomplish a project.
    - Establish priorities and set challenging, achievable goals.
    - Create a plan with specific timelines for completion to achieve the goals.
    - Take initiative to select strategies, resources and/or learning opportunities to accomplish the task(s) in the plan.
    - Identify the success criteria/metrics to determine the effectiveness of the outcome for each goal.
- ACP.10.2 Adapt to organizational changes and expectations while maintaining productive and cooperative relationships with colleagues.
    - Monitor progress/productivity and self-correct during the learning process.
- ACP.10.3 Select and use appropriate technologies to increase productivity.
    - Use appropriate technology tools and resources to create and deliver a product.
- ACP.10.4 Employ leadership skills that build respectful relationships and advance the organization (e.g., recognize and engage individual strengths, plan for unanticipated changes, pursue solutions/improvements).
    - Reflect upon learning (strengths and weaknesses) and use feedback to modify work or improve performance.
    - Persist when faced with obstacles or challenges.

PROFESSIONAL SKILLS: CRITICAL THINKING AND INNOVATION

- ACP.11.1 Identify problems and use strategies and resources to innovate and/or devise plausible solutions.
- ACP.11.2 Explain the process of decomposing a large programming problem into smaller, more manageable procedures.
- ACP.11.3 Implement problem-solving and troubleshooting strategies applicable to software development (for example: debugging).
    - Use relevant criteria to eliminate ineffective solutions or approaches and select those that are plausible; put selected alternatives through trials to determine their helpfulness or benefit.

- Describe problem-solving and trouble-shooting strategies.
- Assess the plausibility of the outcomes (smoke test).

- ACP.11.4 Take action or make decisions supported by evidence and reasoning.
  - Evaluate sources of evidence, the accuracy and relevance of information, and the strengths of arguments.
- ACP.11.5 Transfer knowledge/skills from one situation to another.
  - Apply knowledge and skills in new contexts.

PROFESSIONAL SKILLS: LEGAL AND ETHICAL PRACTICES

- ACP.12.1 Describe current legal issues in the field of computer programming
  - Explain current legal issues in the field of computer science/programming and their implications in the workplace.
- ACP.12.2 Observe laws, rules, and ethical practices in the workplace.
  - Comply with required laws and regulations in the workplace, including employment laws and policies.
  - Apply policies and procedures of the organization based on organizational training(s).
  - Manage and use organizational resources prudently and responsibly.
  - Protect the organization's intellectual and physical property.
- ACP.12.3 Follow industry safety standards in the classroom to maintain a safe work environment.
  - Demonstrate safety standards in the classroom and in the work environment.
  - Apply procedures for reporting unsafe and hazardous conditions in the workplace.

PROFESSIONAL SKILLS: COLLABORATION

- ACP.13.1 Take responsibility for any role on a team and accurately describe and perform the duties of each role, including leadership.
  - Assess project needs and work with a team in a positive manner to create a final project.
  - Build team relationships.
- ACP.13.2 Integrate diverse ideas, opinions, and perspectives of the team and negotiate to reach workable solutions.
  - Contribute personal strengths to a project.
  - Respect the contributions of others.
  - Interact respectfully with co-workers and other professionals in the field.
  - Utilize technologies that promote collaboration and productivity, as appropriate or needed.
- ACP.13.3 Prioritize and monitor individual and team progress toward goals, making sufficient corrections and adjustments when needed.
  - Proactively solicit feedback; accept and show appreciation for constructive feedback.
  - Act upon feedback to achieve team goals.
  - Develop a plan for improving individual participation and group productivity.
- ACP.13.4 Submit high-quality products that meet the specifications for the assigned task.
  - Critique and reflect on individual and collaborative strengths and weaknesses.
- ACP.13.5 Utilize technologies that promote collaboration and productivity.
  - Develop a plan for improving individual participation and group productivity.