CAREER AND TECHNOLOGY EDUCATION
Information Technology

IT.ADV Advanced Placement Computer Science*

Standards ***Indicators***

IT.ADV.10 Demonstrate knowledge of numerical and symbolic computation.
[HCLG>MA{1:EX1.1, 1.2; 2:EX2.2, 2.3 } SC{ 1:EX1.6; also see note below }]

- **IT.ADV.10.01** Describe the problems associated with numerical algorithms (i.e. approximations and accuracy).
- **IT.ADV.10.02** Describe the problems associated with the round-off effect.
- **IT.ADV.10.03** Evaluate expressions using infix notation, prefix notation, postfix notation and boolean algebra.
- **IT.ADV.10.04** Convert expressions between infix, prefix and postfix notations.

IT.ADV.20 Demonstrate proficiency in programming.
[HCLG> MA{ 1:EX1.1, 1.2; 2:EX2.2, 2.3; 3:EX3.2 } } SC{ 1:EX1.6; also see note below } SFS { 1:EX1.2, 1.3; 2:EX2.1, 2.2, 2.3, 2.4 }]

- **IT.ADV.20.01** Write programs using classes.
- **IT.ADV.20.02** Write programs or program segments using arrays, structs, and objects.
- **IT.ADV.20.03** Write programs or program segments using files.
- **IT.ADV.20.04** Write programs or program segments using static representations of stacks and queues.
- **IT.ADV.20.05** Define pointers.
- **IT.ADV.20.06** Compare static and dynamic allocation of memory in the representation of stacks and queues.
- **IT.ADV.20.07** Write programs or program segments using dynamic representations of stacks, queues and linked lists.
- **IT.ADV.20.08** Write programs or program segments using tree structures.
- **IT.ADV.20.09** Compare recursion with iteration.
- **IT.ADV.20.10** Write programs or program segments that implement recursive algorithms.
- **IT.ADV.20.11** Write programs or program segments that implement sequential searches, binary searches and hashing.
- **IT.ADV.20.12** Write programs or program segments that implement bubble, quick and heap sorts.
- **IT.ADV.20.13** In small groups, illustrate a computer language using syntax diagrams and Backus-Naur Form.

IT.ADV.30 Summarize knowledge of algorithms and data structures.
[HCLG>MA{ 1:EX1.1, 1.2; 2:EX2.2; 3:EX3.1, 3.2 } SC{ 1:EX1.6; also see note below }SFS{ 1:EX1.3; 2:EX2.3, 2.4 }]

- **IT.ADV.30.01** Perform simple, best, average and worst case analysis of algorithms.
- **IT.ADV.30.02** Perform Big-O comparisons.
- **IT.ADV.30.03** Define and use arrays, structs, and objects.
- **IT.ADV.30.04** Define and use linked lists.
- **IT.ADV.30.05** Define and use stacks, queues and trees.
- **IT.ADV.30.06** Perform pre-order, in-order and post-order transversals of dynamic data structures.
- **IT.ADV.30.07** Perform insertions and deletions to dynamic data structures.
- **IT.ADV.30.08** Describe, use, and compare various sorting and searching algorithms.
- **IT.ADV.30.09** Describe and use graphs.
- **IT.ADV.30.10** Perform graph transversals.
- **IT.ADV.30.11** Develop and use recursive algorithms.

IT.ADV.40 **Demonstrate knowledge of software methodology and engineering. [HCLG>EN{ 1:EX1.1; 3:EX3.1, 3.3 } MA{ 1:EX1.1, 1.2; 2:EX2.2, 2.3; 3:EX3.2 } SC{ 1:EX1.6; also see note below } SFS{ 1:EX1.1, 1.3; 2:EX2.1, 2.2, 2.3, 2.4; 3: EX3.1, 3.2 }]**

- **IT.ADV.40.01** Define program architecture, modularity, abstraction, information hiding, functional independence, procedural design and reusability.
- **IT.ADV.40.02** Critically evaluate a program design based on the use of modularity, abstraction, information hiding, functional independence and procedural design.
- **IT.ADV.40.03** Describe the preconditions, post-conditions and exceptional conditions in the subprogram specification.
- **IT.ADV.40.04** Use desk checking, walk-throughs and inspections for debugging.
- **IT.ADV.40.05** In groups, describe and develop integration tests for team projects.
- **IT.ADV.40.06** In report form, discuss the concepts of coupling and cohesion and their influence on program design.
- **IT.ADV.40.07** Develop and illustrate test plans including black box and white box tests.

IT.ADV.50 **Summarize knowledge of computer architecture. [HCLG>SFS{ 2:EX2.3, 2.4; 4:EX4.1, 4.2 }]**

- **IT.ADV.50.01** Explain the use of error-correcting codes
- **IT.ADV.50.02** Illustrate the concept of a multilevel machine.
- **IT.ADV.50.03** Working in small groups, report on the function of the various parts of the central processing unit.
- **IT.ADV.50.04** Research and illustrate the function of buses, multiplexers, decoders, arithmetic logic units, shifters, etc. at the conceptual level.
- **IT.ADV .50.05** Research and report on the various types of processors.
- **IT.ADV.50.06** In report form, define the virtual machine**.**
- **IT.ADV.50.07** Research the reasoning behind multilevel machines.
- **IT.ADV.50.08** Research and illustrate the six levels present in most modern computers.
- **IT.ADV.50.09** Research and illustrate the various types of memory.

IT.ADV.60 **Summarize knowledge of social, ethical and professional issues. [HCLG>EN{ 2:EX2.1, 2.2; 3:EX3.1, 3.3 } SC{ 1.EX1.7 } SFS{ 4:EX4.1, 4.2, 4.3; 5:EX5.1, 5.2, 5.4 }]**

- **IT.ADV.60.01** Discuss the issues related to network security (virus, worm, hackers, etc.).
- **IT.ADV.60.02** Describe major issues surrounding security and encryption of sensitive data.

* **High School Core Learning Goals Key**

HCLG	High School Core Learning Goal
EN	English
MA	Mathematics
SC	Science
SFS	Skills for Success
EX	Expectation

Science Note: Programming problems come from many disciplines**.** Thus, algorithms that solve problems or simulate phenomena in science are a part of programming classes. This is another way, AP Computer Science supports the high school core learning goals in science.