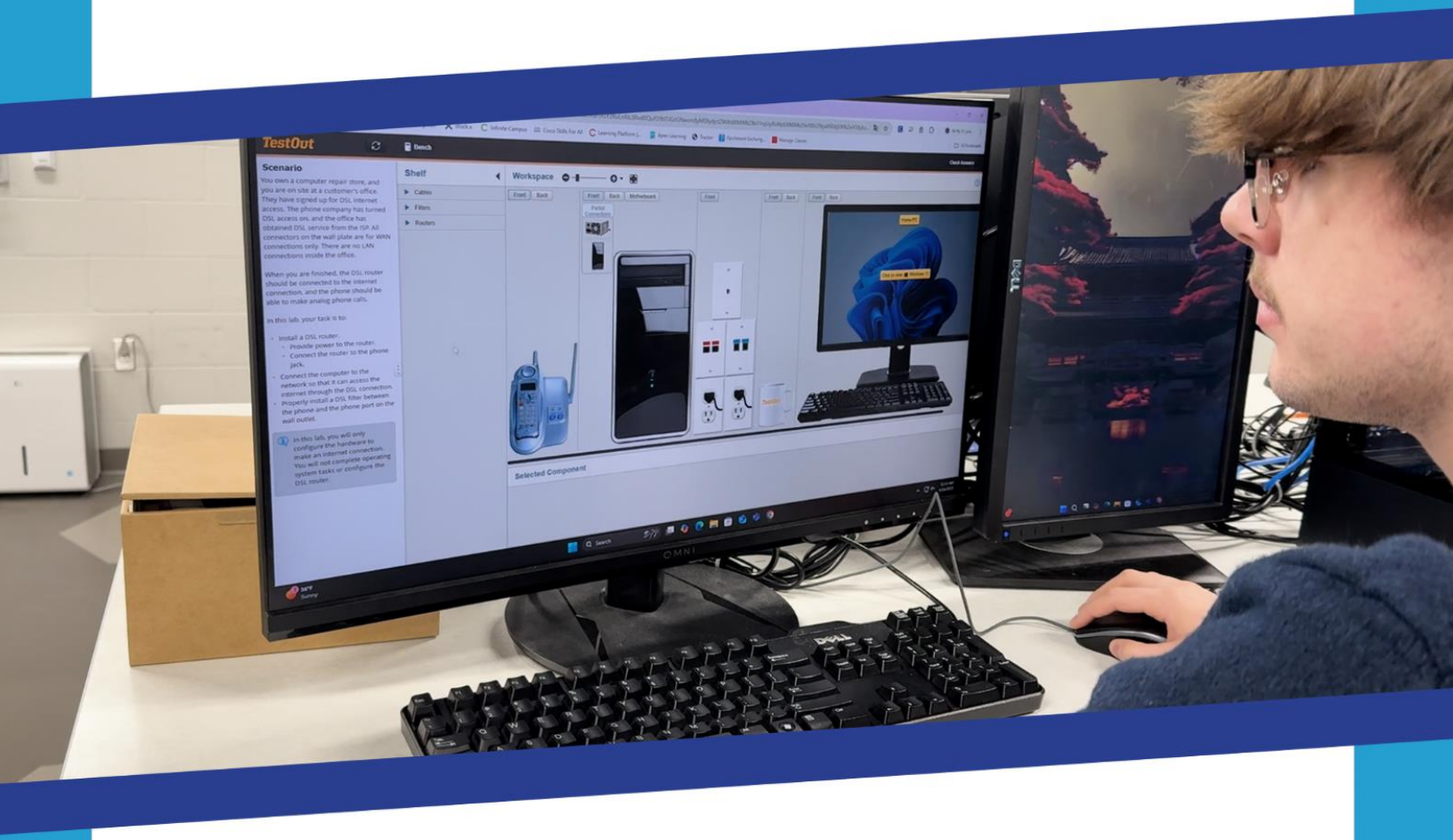
2027-2028
Computer Science Courses



KENTUCKY CTE
EMPOWERED
Today's Career and Technical Education

Kentucky Department of
EDUCATION

COMPUTER SCIENCE EDUCATION COURSES
2027 – 2028

Table of Contents

110101 Computer Hardware and Software Maintenance

This course provides a practical examination of computer hardware systems and client operating systems. Students explore hardware components, system assembly, troubleshooting, repair, and preventative maintenance practices used in industry environments. Instruction includes operating system interfaces, configuration tools, networking components, security practices, and standard operating procedures for system support. Students apply structured troubleshooting methodologies to diagnose and resolve hardware and software issues while following industry safety and documentation standards. Emphasis is placed on system reliability, data protection, operational security, and professional workplace practices. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: .5 – 1

Students will:

1. Identify and explain motherboard components.
2. Identify, install, configure, and upgrade personal computer components.
3. Perform device driver installation, scheduled maintenance, and memory and firmware updates.
4. Identify common tools, basic diagnostic procedures, troubleshooting techniques, and preventative maintenance methods.
5. Explain and apply the troubleshooting process to diagnose and repair common hardware and software problems.
6. Demonstrate an understanding of conversion between binary, decimal, and hexadecimal number systems.
7. Compare and contrast client operating systems and their features.
8. Use multiple user interfaces, including command-line, to perform operating system management tasks, configure, optimize and upgrade the current client operating system and diagnose network connection issues.
9. Use and manage file systems, operating system utilities, backup programs, and optimization tools.
10. Describe the process to install, configure, secure and troubleshoot a basic small or home office network.
11. Identify the fundamental principles of networking and security.
12. Describe and apply appropriate operational procedures, including safety, environmental procedures, good communication skills, and professional behavior.

110102 Help Desk Operations

This course introduces the tools, techniques, and professional practices used to provide effective end-user support in technical help desk environments. Students explore help desk concepts, customer service strategies, structured troubleshooting methodologies, technical writing for end users, needs analysis, facilities management, and help desk software systems. Instruction emphasizes systematic problem solving, documentation practices, communication skills, and the operational procedures necessary to diagnose and resolve user-reported technical issues. Students develop professional competencies in customer interaction, incident management, and service delivery aligned to industry standards. Ethical considerations related to data privacy, security, and responsible technology support are integrated throughout the course. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Define the role of help desk and customer service in an organization.
2. Evaluate help desk technology, tools, and techniques.
3. Identify common support problems, including software tools and features.
4. Identify service technology trends.
5. Demonstrate professional and effective communication skills.
6. Demonstrate team-building strategies.
7. Develop technical training materials and other user documentation to support help desk operations.
8. Demonstrate a methodical approach to the problem-solving process.
9. Apply conflict resolution techniques and skills in customer support.
10. Exhibit positive professionalism with customers and technical writing skills.
11. Demonstrate personal, system, and stress management by way of using self-help tools.
12. Use support performance and reporting tools, call management software, problem resolution software, asset and change management tools, and notification tools for support in additional level two and level three support tools.

110110 Computer Literacy

This course provides an introduction to computer systems and the convergence of technology in today's global environment. Students explore foundational concepts including computer hardware and software, file management, the Internet, electronic communication, social web technologies, green computing, augmented and virtual reality, cybersecurity, and introductory artificial intelligence concepts. Instruction includes the basic use of application, systems, utility, and introductory programming software to support digital productivity and problem solving. Emphasis is placed on computational thinking, responsible technology use, data awareness, and understanding the societal and ethical impacts of computing and emerging technologies. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Describe basic computer functions and use correct computer terminology.
2. Use a course management system.
3. Utilize computer technology as a tool to access, manage, prepare, and present information.
4. Identify trends in information processing and new emerging technologies.
5. Explain the impact of computers upon society, including the effects of social technologies, green computing, dangers of excessive use, and disposal of obsolete equipment.
6. Identify and analyze ethical issues such as copyright, privacy, and security related to computing.
7. Explain the difference between application, programming, system, and utility software.
8. Use a graphical user interface-based operating system to manage files, folders and disks.
9. Use application software packages to prepare basic documents, spreadsheets, databases, and presentations.
10. Describe and explain basic data communications and network technologies and functions.
11. Identify and use basic e-mail and Internet functions and understand their capabilities.
12. Describe globalization and challenges, including technological barriers, electronic payments, and varying cultures.
13. Describe cloud computing and its impact on business and personal systems.
14. Explain the importance of maintaining a good digital identity.
15. Explain how Cellular service differs from Internet service.
16. Explore the impact of AI on society.

110201 Introduction to Programming

This course focuses on the design, development, and implementation of computer programs to solve problems and support system functionality. Instruction includes software design principles, programming languages, structured program development, and systematic troubleshooting. Students are introduced to fundamental programming concepts using an industry-recognized or emerging programming language. Course content includes data types, control structures, basic data structures, error handling, modular programming, information and file processing, and characteristics unique to the language used in the course. Emphasis is placed on computational thinking, algorithmic design, and problem solving as students develop and analyze programs. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Demonstrate knowledge of the program development life cycle.
2. Design, develop, compile, debug, test, run, and document programs in the language studied.
3. Design and develop programs using operators and assignments.
4. Design and develop programs that properly use variables, constants, data types, and objects.
5. Design and develop programs that use sequence, selection, and repetition structures.
6. Design and develop programs that use simple data structures.
7. Design and develop programs that use effective error and exception handling.
8. Design and develop programs that implement user-defined methods and modular programming.
9. Design and develop programs that implement file processing.
10. Design and develop programs that implement fundamental features that are unique to the language studied.
11. Design and develop programs using object-oriented programming features, if applicable to the language studied.
12. Explain how algorithms are used to produce artificial intelligence (AI).
13. Evaluate and critique the effectiveness and efficiency of code written.

110204 Data Science Principles

This course introduces foundational concepts in data science through the use of word processing, spreadsheet, database, and presentation software to analyze and communicate information. Students collect, organize, visualize, and interpret data to solve technology and business problems using emerging technologies and industry-relevant tools. Instruction emphasizes data representation, pattern recognition, structured problem solving, and ethical data practices. Students develop an understanding of how computing systems process and represent data and apply computational thinking to draw evidence-based conclusions and communicate findings effectively. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Use a productivity software package to create, edit, print, and save documents.
2. Use productivity tools such as spelling and grammar.
3. Apply formatting features such as font, color, margins, headers, and footers.
4. Use tools such as cut, copy, and paste within a document and between documents.
5. Create HTML file formats for web publishing.
6. Create new documents using templates and wizards.
7. Use a word processing program to insert and use table features.
8. Use a word processing program to insert and use table column features.
9. Insert pictures and Clipart into word-processing documents.
10. Use a spreadsheet package to create common business reports and budgets.
11. Use mathematical formulas and common statistical, date, financial, and logical functions.
12. Make formatting changes to a worksheet, including column width, row height, cell, and table formatting.
13. Use autofill to copy and paste formulas and repeat patterns.
14. Create effective charts, including bar, line, and pie charts, to accompany business reports.
15. Use a relational database management program to create tables, queries, forms, reports, and labels.
16. Use query features to extract information from a database using simple and compound conditions.
17. Use the relationship feature to join tables in a database and obtain information from multiple tables.
18. Plan and create an electronic slide show presentation using a presentation software package.
19. Use timing, transition, and animation features to enhance a presentation.

110211 Introduction to Database Design

This course provides an overview of database and database management system concepts, including internal design models, normalization, network and relational data models, development tools, and database applications. Students examine how structured data is organized, stored, retrieved, and maintained within computing systems. Instruction emphasizes computational thinking, data representation, and systematic design as students model data structures and develop database solutions to meet user and organizational needs. Students analyze data integrity, security, and ethical considerations related to data management. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Define a database and its uses.
2. Describe the difference between traditional files and databases.
3. Define a database management system (DBMS) and describe the services a DBMS provides to its users.
4. Identify and describe the main features of hierarchical, network, and relational database models.
5. Demonstrate an understanding of the difference between logical and physical design.
6. Model a realistic business application using a technology-independent data model.
7. Design and implement a database using the relational model, with an emphasis on data integrity and security.
8. Define and use the normalization process to further refine the relational table definitions.
9. Demonstrate an understanding of the database administration function.
10. Define and be able to use data definition language, data manipulation language, and instructions that apply relational algebra.
11. Demonstrate an understanding of distributed database systems.
12. Evaluate and select an appropriate DBMS for a given application.

110213 Design for the Internet

This course introduces foundational principles of digital design and computer graphics with an emphasis on creating visual content for web-based environments and interactive media. Students explore graphic design concepts, layout principles, image optimization, multimedia integration, and user experience considerations for online platforms. Instruction emphasizes structured design processes, problem solving, and the effective use of digital tools to develop visually engaging and functional web content. Students examine how computing systems render and display graphics across devices while considering accessibility, usability, and ethical use of digital media. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Identify the principles of communication through the visual medium using text, still imagery and video technology.
2. Explain copyright laws affecting digital graphics, including images and image use.
3. Identify the purpose of an audience, storyboarding, and audience needs for preparing images.
4. Explain the design process for various forms of digital media.
5. Identify considerations of designing for a specific audience, including paid customers.
6. Analyze and evaluate digital media content for audience, purpose and design techniques.
7. Identify trends in the use and creation of digitally generated media.
8. Explain the key elements of drawing and painting.
9. Explain image resolution, image size, and image file format for the web, video, and print.
10. Demonstrate effective message composition and design using industry-standard design elements and principles.
11. Explain the principles of image composition.
12. Explain digital typography.
13. Differentiate between typeface and font.
14. Demonstrate digital camera and scanner operation.
15. Define digital image terminology.
16. Explain image and editing layers.
17. Demonstrate importing, exporting, organizing, and saving digital graphic files.
18. Manipulate image selections and measurements.
19. Use digital graphic editing software guides and rulers.
20. Transform digital images using editing applications.
21. Adjust or correct the tonal range, color, or distortions of an image using editing applications.
22. Explain retouching and blending images.
23. Explain and apply digital image editing filters.
24. Prepare images for web, print, and video.
25. Identify career and entrepreneurial opportunities in digital graphics technology.

110220 Object-Oriented Programming I

This course introduces students to advanced programming concepts using an object-oriented programming language selected to meet student and program needs. Instruction emphasizes object-oriented design principles including abstraction, encapsulation, inheritance, and polymorphism as students develop structured and reusable software solutions applicable to modern and emerging technologies. Course content includes data types, control structures, arrays, user interface development, modular programming, error handling, and foundational data structures. Students apply computational thinking and algorithmic design to analyze problems and construct efficient, maintainable programs used in software development and technology-driven systems. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Design, develop, compile, debug, test, run and document programs.
2. Demonstrate knowledge of the program development life cycle.
3. Design and develop programs using operators and assignments.
4. Design and develop programs using primitive data types.
5. Design and develop programs using a variety of data types.
6. Design and develop programs using sequences, selection, and repetition structures.
7. Design and develop programs using single and multi-dimensional arrays.
8. Design and develop programs using arrays, lists and tuples.
9. Utilize arrays
10. Design and develop programs using effective error and exception handling.
11. Design and develop programs using object-oriented programming features, including defining classes, instantiating objects, and using arrays of objects.
12. Design and develop programs implementing user-defined methods and modular programming.
13. Explain how algorithms are used to produce artificial intelligence (AI).
14. Design and develop programs using method overloading.
15. Design and develop programs using inheritance, encapsulation, and polymorphism.
16. Design and develop programs using simple GUI components.
17. Evaluate and critique the effectiveness and efficiency of code.

110221 Object-Oriented Programming II

This course provides students with advanced study in the design and development of complex object-oriented applications. Instruction extends object-oriented principles through deeper exploration of inheritance, polymorphism, abstraction, multithreading, recursion, file processing, and application architecture. Teachers select the programming language(s) most appropriate to support student learning and industry alignment. Students design, implement, and evaluate scalable software solutions applicable to modern computing environments and emerging technologies, including mobile and distributed systems. Emphasis is placed on computational thinking, algorithmic efficiency, system reliability, and structured problem solving. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Design and develop programs that use advanced GUI Components.
2. Design and develop programs that use input and output streams, including character and binary streams.
3. Demonstrate knowledge of advanced concepts and associated definitions.
4. Design and code applications using advanced data types and structures.
5. Design and develop programs that use concurrency.
6. Design, develop, compile, debug, test, run and document advanced programs in the language.
7. Design and develop programs using polymorphism, inheritance, and overloading.
8. Design and develop programs that incorporate other advanced features.
9. Examine and evaluate the strengths and weaknesses of the language(s).
10. Demonstrate error-checking and error-handling.
11. Implement input validation and processing.
12. Evaluate and critique the effectiveness and efficiency of code.

110222 Cyber Literacy I

Cyber Literacy I is a hands-on course that builds a strong foundation in cybersecurity and computing systems for high school students. The course introduces cybersecurity concepts by integrating robotics, programming, electricity, networking, and interdisciplinary perspectives. Students explore the opportunities, threats, responsibilities, and legal considerations associated with operating in cyberspace. Instruction emphasizes foundational principles of electricity, programming, networking, and system interactions while developing critical thinking and structured problem-solving skills. Students examine ethical responsibilities, data protection, and operational security practices that support secure computing environments. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Apply the fundamentals of electricity, including the basic movement of electrons and experiments that include chemistry, circuitry, and magnetism.
2. Apply BASIC Programming including basic coding essentials through flowcharts, the use of simple programming languages, and the use of simple programming tasks.
3. Engage with a microcontroller as a platform for learning robotics fundamentals.
4. Assemble robots to perform various functions through the implementation of sensors.
5. Apply programming knowledge including the use of autonomous devices, the use of programming components such as conditional and unconditional loops, subroutines, and variable manipulation.
6. Relate electrical components like LEDs, piezo-crystal elements, infrared light, and tactile sensors to cyber.
7. Illustrate real-world applications and implications of computers and the Internet in our society today.
8. Deliberate the historical and societal context of cyber.
9. Engage in a lab and project-driven lesson regarding learning about cyberspace.
10. Engage in a lab and project-driven lesson regarding the ethical concerns about online behavior, cyberbullying, and cybersecurity.
11. Engage in a lab and project-driven lesson regarding the ethical concerns of designing autonomous devices and artificial intelligence.

110223 Cyber Literacy II

Cyber Literacy II is a project-driven course that expands students' understanding of cybersecurity and cyberspace through systems engineering and interdisciplinary analysis. Building upon foundational cyber concepts, students examine system architecture, networked environments, and integrated technologies that support secure operations. Instruction emphasizes systems thinking, engineering design processes, schematic interpretation, and structured documentation practices. Students develop flowcharts and technical designs to guide system builds, analyze interactions among hardware and software components, and apply principles of science, engineering, technology, and mathematics within cybersecurity contexts. Ethical considerations, operational security, and responsible decision-making in cyber environments are reinforced throughout the course. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Apply systems engineering to build a variety of multi-component projects.
2. Employ flowcharts to discuss data flow and pseudo-code.
3. Engage in a variety of labs and project-driven lessons that prompt learning regarding robotics.
4. Combine components onto a platform to design autonomous systems while considering the real-world impact of the systems and how they impact the students' environment.
5. Synthesize content with issues including privacy, security, and technology.
6. Synthesize content with issues including search warrants, digital media, and the requirements to obtain a search warrant.
7. Synthesize content with issues including cyberbullying and real-world examples of the implications of cyberbullying.
8. Defend a position in debates on national security.
9. Engage in a variety of labs and project-driven lessons that prompt learning regarding the impact and relationship between expectations of privacy and security.

110224 Cyber Science

Cyber Science is an innovative, project-driven course that integrates science, technology, engineering, mathematics, and liberal arts through a systems-level approach to problem solving. Students apply robotics and computer science concepts within interdisciplinary contexts to analyze complex challenges and design integrated solutions. Instruction includes networking and security fundamentals, programming basics, foundations of computer science, ethics and societal issues, and introductory artificial intelligence concepts. Students examine how computing and AI systems function, interact, and impact individuals and communities while developing responsible digital citizenship. Emphasis is placed on computational thinking, systems design, and structured problem solving across disciplines. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Apply programming basics, including developing programming skills through a progression of robot activities.
2. Apply foundations of computer science, including Boolean logic, variables, flow charts, data structures, and sorting using robot applications.
3. Apply networking and security foundations including showcasing the structure of networks as well as the vulnerabilities.
4. Demonstrate the need for security through emphasizing man-in-the-middle attacks, cryptography, and stenography.
5. Utilize artificial intelligence, including applying the concepts of heuristics.
6. Utilize sensors to read input in order to produce the desired output through various robot projects.
7. Explore the historical, ethical, and societal impacts of cyber.
8. Defend a position in debates regarding cyber.

110225 Computer Science Fundamentals

Computer Science Fundamentals is a project-driven, application-based course that engages students in an immersive exploration of the breadth of computer science. Students design and create programs, construct simple circuits, and examine how hardware and software systems interact within computing environments. Instruction emphasizes computational thinking, algorithmic design, structured problem solving, and data representation as students develop and analyze solutions to real-world challenges. Students explore the societal and ethical impacts of computer science, including responsible technology use and the influence of computing on individuals and communities. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Develop algorithms as solutions to problems while discovering the importance of understanding problems in order to develop efficient step-by-step solutions.
2. Explore the reasons for using computers to execute problem solutions.
3. Translate algorithms to a language computers can understand.
4. Explore various major data manipulation and processing structures used by computers.
5. Explore and utilize structures to design algorithms to solve problems.
6. Apply computer architecture, including using a Raspberry Pi platform, to learn how computer hardware provides a powerful platform on which to run the software.

110226 Project-Based Programming

Project-Based Programming is an application-driven course designed for students interested in developing innovative solutions through programming. Students apply computer science fundamentals to design, research, and develop original software applications that address authentic problems. Projects may be completed individually or collaboratively and require sustained inquiry, planning, and execution. Instruction emphasizes computational thinking, algorithmic design, structured research, time management, scope development, and collaborative problem solving. Students demonstrate proficiency in software development practices, documentation, testing, and iterative improvement while developing leadership and professional skills. The teacher serves as a facilitator, guiding students through the project development lifecycle and encouraging ownership of learning. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Manage and modify the scope to ensure success.
2. Apply time management skills to complete projects to specifications effectively.
3. Use the programming language of choice to develop a project of choice.
4. Demonstrate continuous improvement through feedback from one-on-one meetings each week and regular project updates with the facilitator.
5. Use appropriate and effective research skills.
6. Demonstrate proficiency in computer science fundamentals and design patterns.
7. Architect a project for the semester.
8. Demonstrate the use of version control through Git.
9. Demonstrate manipulation of search terms in Google to obtain valid and useful results.

110230 Cybersecurity

Cybersecurity introduces the tools, concepts, and practices used to protect computing systems, networks, and data. Students examine how individuals and organizations share computing resources while safeguarding privacy and maintaining operational security. The course develops students' knowledge of ethical computing behavior and responsible decision-making in digital environments. Instruction includes the core components of cybersecurity, including risk management, threat identification, vulnerability analysis, prevention strategies, detection methods, and mitigation techniques. Students analyze how hardware, software, and networked systems interact within secure computing environments and evaluate strategies used to defend against cyber threats. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Identify the goals, objectives and purposes of cybersecurity
2. Research the history of the Internet and its impact on government, society and business.
3. Describe the concepts of malware attack vectors
4. Mitigate threats by remaining abreast of industry information
5. Explain basic security concepts
6. Identify issues that affect physical and remote access device security
7. Explain data and privacy encryption issues related to using technology
8. Compare and contrast physical security controls
9. Explain the purpose of various network access control models
10. Identify common network vulnerabilities, threats, and risks
11. Explain the functions and applications of various network devices
12. Describe the need for security safeguards
13. Demonstrate the components of cybersecurity and the role each plays in preventing and detecting attacks.
14. Assess the role of strategy and policy in determining the success of information security.
15. Contrast the various approaches to security training and formulate a simple training agenda.
16. Evaluate the trends and patterns that will determine the future state of cybersecurity.

110231 AP Cybersecurity

AP Cybersecurity is a full-year course that introduces students to foundational principles and advanced practices in cybersecurity at a level comparable to an introductory college course. Students examine the evolving threat landscape and apply defense-in-depth strategies to protect systems, networks, devices, applications, and data. Students analyze vulnerabilities, attacks, mitigation techniques, and detection methods while developing skills in risk management, cryptography, access control, system hardening, and incident response. Instruction emphasizes hands-on learning, real-world scenarios, and the evaluation of security frameworks. Students engage with tools, technologies, and automated processes to design, implement, and refine secure system configurations and defensive solutions, strengthening computational thinking and problem-solving skills. Learning experiences may include the use of programming and emerging technologies to support cybersecurity practices. The course also explores the ethical, legal, and societal implications of cybersecurity. Participation in the Technology Student Association (TSA) or SkillsUSA will greatly enhance instruction.

Recommended Grade Level: 11 – 12

Recommended Credit: 1

Students will:

1. Describe the evolving cyber threat landscape and the goals and motives of different types of threat actors.
2. Identify key concepts in cybersecurity such as confidentiality, integrity, and availability (CIA Triad).
3. Explain how common vulnerabilities can be exploited and how they impact systems and organizations.
4. Distinguish between physical, administrative, and technical security controls and provide examples of each.
5. Demonstrate proper implementation of user authentication and access control methods (e.g., passwords, MFA, RBAC).
6. Explain fundamental cryptographic concepts including encryption, hashing, digital signatures, and certificates.
7. Analyze the role of firewalls, antivirus, and intrusion detection/prevention systems in securing networks.
8. Identify and respond to various types of malware, including viruses, worms, trojans, ransomware, and spyware.
9. Configure secure settings for user accounts, file permissions, and system services in a virtual lab environment.
10. Conduct basic vulnerability scans and interpret results to identify potential system weaknesses.
11. Explain the importance of patch management and software updates in maintaining security posture.
12. Evaluate network traffic using tools like Wireshark to detect suspicious behavior or policy violations.
13. Describe security best practices for cloud computing and virtualization environments.
14. Explain the legal and ethical considerations in cybersecurity, including data privacy and responsible disclosure.
15. Use risk management frameworks to assess threats and propose mitigation strategies.
16. Outline the stages of incident response and describe actions taken in detection, containment, eradication, and recovery.

17. Research current trends and emerging technologies in cybersecurity, such as AI, blockchain, and IoT security.
18. Compare and contrast cybersecurity career roles (e.g., SOC analyst, penetration tester, compliance officer).

110251 Computational Thinking

Computational Thinking develops students' understanding of programming and logic by applying structured problem-solving strategies used in computing systems and emerging technologies. Students design language-independent solutions to solve technology-based problems through algorithmic thinking and systematic analysis. Instruction includes foundational design principles such as variables, control structures, data structures, abstraction, decomposition, and the use of command-line and object-oriented programming concepts. Emphasis is placed on developing clear, efficient, and scalable solutions that support modern computing environments. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Demonstrate an understanding of elementary logic, truth tables, and Boolean algebra.
2. Demonstrate programming style best practices.
3. Illustrate the flow of a program.
4. Illustrate concepts using one or more programming languages.
5. Explain the implications of file processing.
6. Describe the steps addressed in the design of a program to solve the state problem.
7. Explain how algorithms are used to produce artificial intelligence (AI)
8. Describe the principles of object-oriented programming.
9. Develop algorithms with an increasing degree of complexity using structured programming techniques such as sequence, selection, and repetition.
10. Use fundamental data types and data structures such as integers, reals, characters, strings, Booleans, and one - and two - dimensional arrays.
11. Analyze the binary representation of data.
12. Use modular programming.

110302 Management of Support Services

This course provides in-depth study of the management, coordination, and oversight of technology support services within organizational environments. Students examine service delivery models, help desk operations, resource allocation, policy development, documentation standards, and operational procedures aligned to evolving industry needs and emerging technologies. Instruction emphasizes systems thinking, structured problem solving, data management, and professional leadership practices within information and support services. Students analyze service workflows, evaluate technology infrastructure needs, and apply best practices related to security, reliability, and ethical technology management. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs.

Recommended Grade Level: 11 – 12

Recommended Credit: 1

Students will:

1. Determine purpose and goals using a project management method.
2. Define, Design, Develop, Deploy, Reflect, Redesign, and Present utilizing presentation software visualizing the process.
3. Determine roles, tasks, and calendars.
4. Utilize Software packages for project management (MS Project, Excel, Visio, Dread-Spark, Prezi).
5. Utilize and define appropriate terminology.
6. Present information in a technical report.
7. Publish information presented to an advisory board member.
8. Identify potential employment barriers for non-traditional groups and ways to overcome the barrier.
9. Research potential barriers to placing information in a spreadsheet.
10. Present information to school principals and peers.
11. Synthesize the information collecting, producing a product that could help overcome the barrier with non-traditional groups.

110701 AP Computer Science A

AP Computer Science A introduces students to computer science through rigorous object-oriented programming and problem solving, primarily using the Java programming language. Fundamental topics include the design of solutions to complex problems, the implementation and analysis of algorithms, and the use of data structures to organize and process information. Students apply principles of abstraction, modularity, and efficient design to develop and refine scalable software solutions while strengthening computational thinking, algorithmic analysis, and problem-solving skills. Instruction emphasizes the development, testing, and evaluation of object-oriented programs using programming and related tools. The course also explores the ethical and societal impacts of computing systems. Programming is defined by the K-12 Computer Science Framework as the process of analyzing problems and designing, developing, testing, and refining computational solutions. Participation in the Technology Student Association (TSA) or SkillsUSA will greatly enhance instruction.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Design, implement and analyze solutions to problems.
2. Use and implement commonly used algorithms.
3. Develop and select appropriate algorithms and data structures to solve new problems.
4. Write solutions fluently in an object-oriented paradigm.
5. Write, run, test, and debug solutions in the Java programming language, utilizing standard Java library classes and interfaces from the AP Java Subset.
6. Read and understand programs consisting of several classes and interacting objects.
7. Read and understand a description of the design and development process leading to such a problem.
8. Understand the ethical and social implications of computer use.

110710 Introduction to Computer Science

Introduction to Computer Science is designed to introduce students to the breadth of the field through an exploration of engaging and accessible topics. Rather than focusing exclusively on specific software tools or programming languages, the course emphasizes the conceptual foundations of computing and helps students understand how and why particular tools, languages, and emerging technologies are used to solve problems. The course develops computational practices including algorithm development, structured problem solving, and programming within contexts relevant to students' lives. Students are introduced to topics such as interface design, limits of computing systems, and societal and ethical implications of computing and emerging technologies. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Analyze the characteristics of hardware components to determine the applications for which they can be used.
2. Use appropriate tools and methods to execute Internet searches that yield requested data.
3. Evaluate the results of web searches and the reliability of information found on the Internet.
4. Explain the differences between tasks that can and cannot be accomplished with a computer.
5. Analyze the effects of computing on society within economic, social, and cultural contexts.
6. Communicate legal and ethical concerns raised by computing innovation.
7. Explain the implications of communication as data exchange.
8. Name and explain the steps they use in solving a problem.
9. Solve a problem by applying appropriate problem-solving techniques.
10. Express a solution using standard design tools.
11. Determine if a given algorithm successfully solves a stated problem.
12. Create algorithms that meet specified objectives.
13. Explain the connections between binary numbers and computers.
14. Summarize the behavior of an algorithm.
15. Compare the tradeoffs between different algorithms for solving the same problem.
16. Explain the characteristics of problems that cannot be solved by an algorithm.
17. Use appropriate algorithms to solve problems.
18. Design, code, test, and execute a program that corresponds to a set of specifications.
19. Select appropriate programming structures.
20. Locate and correct errors in a program.
21. Explain how a particular program functions.
22. Justify the correctness of a program.
23. Create programs with practical, personal, and/or societal intent.
24. Describe the features of appropriate data sets for specific problems.
25. Apply a variety of analysis techniques to large data sets.
26. Use computers to find patterns in data and test hypotheses about data.
27. Compare different analysis techniques and discuss the tradeoffs among them.

28. Justify conclusions drawn from data analysis.
29. Describe ways in which computing enables innovation.
30. Explain how algorithms are used to produce artificial intelligence (AI)
31. Discuss the ways in which innovations enabled by computing affect communications and problem-solving.
32. Analyze how computing influences and is influenced by the cultures for which they are designed and the cultures in which they are used.
33. Analyze how social and economic values influence the design and development of computing innovations.
34. Discuss issues of equity, access, and power in the context of computing resources.
35. Communicate legal and ethical concerns raised by computational innovations.
36. Discuss privacy and security concerns related to computational innovations.
37. Explain the positive and negative effects of technological innovations on human culture.

110711 AP Computer Science Principles

AP Computer Science Principles introduces students to the breadth of computer science through computational thinking, abstraction, data, algorithms, programming, the Internet, and the societal impacts of computing. Students design and evaluate solutions to problems and develop computational artifacts using algorithms, programming, and emerging technologies while using data to discover patterns and generate insights. The course emphasizes collaboration, creativity, and ethical responsibility as students explore how computing systems and innovations impact society. Instruction incorporates multiple levels of abstraction and modeling to help students understand how computing systems operate and scale. Students engage in the design, development, evaluation, and refinement of computational solutions, strengthening skills in algorithmic reasoning, data analysis, abstraction, and problem-solving. Programming is defined by the K-12 Computer Science Framework as the process of designing, developing, testing, and refining computational solutions. Participation in the Technology Student Association (TSA) or SkillsUSA will greatly enhance instruction.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Use computing tools and techniques to create artifacts.
2. Collaborate in the creation of computational artifacts.
3. Analyze computational artifacts; use computing tools and techniques for creative expression.
4. Use programming as a creative tool.
5. Describe the combination of abstractions used to represent data.
6. Explain how binary sequences are used to represent digital data.
7. Develop and abstraction.
8. Use multiple levels of abstraction in computation.
9. Use models and simulations to raise and answer questions.
10. Use computers to process information to gain insight and knowledge.
11. Collaborate when processing information to gain insight and knowledge.
12. Communicate insight and knowledge gained from using computer programs to process information.
13. Use computing to facilitate exploration and the discovery of connections in information.
14. Use large data sets to explore and discover information and knowledge.
15. Analyze the considerations involved in the computational manipulation of information.
16. Develop an algorithm designed to be implemented to run on a computer.
17. Express an algorithm in a language.
18. Appropriately connect problems and potential algorithmic solutions.
19. Evaluate algorithms analytically and empirically.
20. Explain how programs implement algorithms.
21. Use abstraction to manage complexity in programs.
22. Evaluate a program for correctness.
23. Develop a correct program.
24. Collaborate to solve a problem using programming.
25. Employ appropriate mathematical and logical concepts in programming.
26. Explain the abstractions on the Internet and how the Internet functions.
27. Explain the characteristics of the Internet and the systems built on it.
28. Analyze how characteristics of the Internet and the system built on it influence their use.

29. Connect the concern of cybersecurity with the Internet and the systems built on it.
30. Analyze how computing affects communication, interaction, and cognition.
31. Collaborate as part of a process that scales.
32. Connect computing with innovations in other fields.
33. Analyze the beneficial and harmful effects of computing.
34. Connect computing within the economic, social, and cultural context.

110752 Applied Computer Science Projects

This course provides in-depth study of specialized or emerging topics within computer science aligned to the career major and responsive to evolving industry needs and emerging technologies. Instruction emphasizes computational thinking, algorithmic design, data representation, and structured problem solving within authentic computing contexts. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. **Course content and objectives must align with Computer Science standards upon approval from the Computer Science Consultant by submitting a pathway modification request.**

Prerequisite: Successful completion of at least (4) four credits in a Computer Science pathway.

Recommended Grade Level: 11 – 12

Recommended Credit: 1

Students will:

1. Complete tasks defined by the teacher and approved by the Computer Science Consultant in the Office of Career and Technical Education.

110801 Web Page Development

This course introduces web page development through the use of HTML and CSS. Students use text and web editors to create structured web documents incorporating layout design, multimedia elements, tables, forms, and responsive page formatting. Instruction emphasizes W3C web design principles, accessibility standards, and industry best practices. Students examine how web technologies function within modern and emerging technology environments while applying design principles to create user-centered digital content. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Plan the layout of a website.
2. Use HTML (Hypertext Markup Language).
3. Use CSS (Cascading Style Sheets).
4. Create lists and tables in an organization's content.
5. Use HTML and CSS in page layout.
6. Create web forms.
7. Use multimedia in the creation of a website (such as images, sound, and video).
8. Publish web pages to a website.

110804 Website Design and Production

This course introduces website production processes with emphasis on design, layout, navigation, usability, and interactivity within digital environments. Students apply industry-relevant web production software and development tools to plan, design, and produce functional and visually effective websites. Instruction emphasizes user-centered design, accessibility standards, responsive layout principles, and integration of multimedia elements within modern and emerging technology environments. Students evaluate website functionality, performance, and user experience as part of the production process. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Utilize principles of graphic and content creation for online media.
2. Use fundamental online graphic design principles, including appropriate interactivity, content-sensitive navigation schemes and user interface criteria.
3. Select task-appropriate software tools.
4. Utilize website accessibility.
5. Utilize website implementation and hosting.
6. Edit and enhance digital video images using state-of-the-art software.

110809 JavaScript

This course provides students with an overview of the JavaScript scripting language used to develop interactive and dynamic web applications. Instruction includes coding, testing, and debugging JavaScript programs; working with variables, operators, and data types; and using control structures, objects, pattern matching, and application scripts. Students apply JavaScript to create dynamic web pages, manage user input, and control the behavior of forms, buttons, and other interface elements within modern and emerging technology environments. Emphasis is placed on computational thinking, algorithmic design, and structured problem solving in web-based systems. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Design, develop, compile, debug, test, run, and document programs in the JavaScript language.
2. Design and develop programs using operators and assignments.
3. Design and develop programs using a variety of data types.
4. Demonstrate the input and output processes in JavaScript.
5. Design and develop programs using sequence, selection, and repetition structures.
6. Demonstrate pattern matching using JavaScript.
7. Demonstrate JavaScript objects.
8. Demonstrate the ability to write JavaScript application scripts.
9. Evaluate and critique the effectiveness and efficiency of code.

110821 App Development with Swift

App Development with Swift provides students with a foundation in Swift programming, UIKit, and networking through hands-on labs and guided projects aligned to modern and emerging technology environments. Throughout the course, students complete structured projects that simulate real-world app development workflows and industry practices. Students design and implement application features, manage user interfaces, and integrate networking components while applying software development principles used in professional environments. In the final unit, students design, prototype, and architect an original app of their own creation. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Explain the basics of data, operators, and control flow in Swift.
2. Explain the basics of documentation, debugging, Xcode, building and running an app, and Interface Builder.
3. Apply this knowledge to create a simple flashlight app.
4. Explore Swift strings, functions, structures, collections, and loops.
5. Explore UIKit and how to display data using Auto Layout and stack views.
6. Practice this knowledge to build a word-guessing game app.
7. Discover how to build simple workflows and navigation hierarchies using navigation controllers, tab bar controllers, and segues.
8. Examine two powerful tools in Swift, optionals and enumerations.
9. Apply this knowledge into practice with a personalized survey that reveals a response to the user.
10. Discover scroll views, table views, and building complex input screens.
11. Explore how to save data, share data with other apps, and work with images in the user's photo library.
12. Practice these skills with a task-tracking app that allows the user to add, edit, and delete items in a familiar table-based interface.
13. Customize the task-tracking app to keep track of any information.
14. Determine how to use animations, concurrency, and how to work with the web.
15. Apply this knowledge with a customizable menu app that displays the available dishes from a restaurant and allows the user to submit an order.
16. Use a web service that allows students to set up a menu with customized menu items and photos.
17. Design, prototype, and architect a project of their own design.
18. Build this project independently.

110901 Introduction to Networking Concepts (non-vendor)

This course introduces technical, non-vendor-specific networking concepts, including network technologies, transmission media, topologies, devices, management tools, and security practices. Students examine how network infrastructure supports communication within modern and emerging technology environments. Instruction provides foundational skills in installing, configuring, operating, maintaining, and troubleshooting basic network systems. Students analyze network performance, apply structured problem-solving strategies, and evaluate security considerations within networked environments. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Differentiate between network protocols in terms of routing, addressing schemes, interoperability, and naming conventions.
2. Identify addressing format, schemes, technologies and required settings for connectivity, including classful/classless address ranges, public/private addressing, and subnetting.
3. Identify the common ports associated with TCP (Transmission Control Protocol) and UDP (User Datagram Protocol).
4. Identify the basic standards of WAN and remote access technologies.
5. Categorize standard cable types and their properties.
6. Explain logical and physical network structures, topologies, and characteristics.
7. Identify the basic attributes, purposes, and functions of network components, including wireless technologies.
8. Identify the functions of specialized network devices such as multilayer switches, load balancers, proxy, DNS servers, and CSU/DSU.
9. Plan and implement a basic wired and wireless network.
10. Explain the function of each layer of the OSI and TCP/IP models.
11. Identify types of configuration management documentation such as network diagrams, wiring schematics, and configurations.
12. Summarize and explain different methods and rationales for network performance.
13. Diagnose a network problem using a systemic approach, identify the appropriate tools, select an appropriate course of action to resolve the problem, and document the solution.
14. Select the appropriate hardware and software tools to test, scan, and analyze network connectivity and performance.
15. Identify and explain common methods to ensure network security, including antivirus software, user authentication, and firewall setup.
16. Identify issues that affect physical and remote access device security.

110902 Network Fundamentals/Cisco I

This course introduces the architecture, structure, functions, components, and models of the Internet and other computer networks. Students examine foundational networking principles including IP addressing, Ethernet concepts, media, and network operations within modern and emerging technology environments. Instruction develops skills in building simple local area networks (LANs), performing basic configuration of routers and switches, and implementing IP addressing schemes. Students analyze network performance and apply structured troubleshooting strategies to support reliable and secure communication systems. Students apply command-line configuration and scripting techniques to configure, test, and troubleshoot network devices and infrastructure. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Describe the devices and services used to support communications in data networks and the Internet.
2. Describe the role of protocol layers in data networks.
3. Describe the importance of addressing and naming schemes at various layers of data networks in IPv4 and IPv6 environments.
4. Design, calculate and apply subnet masks and addresses to fulfill given requirements in IPv4 and IPv6 networks.
5. Explain fundamental Ethernet concepts such as media, services, and operations.
6. Design a simple Ethernet network using routers, switches, cables, connectors and other hardware.
7. Demonstrate the Cisco command-line interface (CLI) commands to perform basic router and switch configurations.
8. Utilize common network utilities to verify small network operations and analyze data traffic.

110903 Routing Protocols and Concepts/Cisco II

This course focuses on switching technologies and router operations that support small- to medium-sized business networks, including wireless local area networks (WLANs) and foundational security concepts. Students examine routing protocols, network segmentation, and traffic management within modern and emerging technology environments. Instruction emphasizes network configuration, troubleshooting methodologies, identification and mitigation of LAN security threats, and the configuration and security of wireless networks. Students implement routing strategies, analyze network performance, and apply structured problem-solving techniques to support secure and reliable network operations. Students apply command-line configuration and scripting techniques to configure, secure, and troubleshoot routing and switching infrastructure. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Describe basic switching concepts and the operation of Cisco switches.
2. Describe enhanced switching technologies such as VLANs, VLAN Trunking Protocol (VTP), Rapid Spanning Tree Protocol (RSTP), Per VLAN Spanning Tree Protocol (PVSTP), and 802.1q.
3. Troubleshoot basic operations of a small switched network.
4. Describe the purpose, nature, and operations of a router, routing tables, and the route lookup process.
5. Configure and troubleshoot static routing and default routing.
6. Describe how VLANs create logically separate networks and how routing occurs between them.
7. Describe dynamic routing protocols, distance vector routing protocols, and link-state routing protocols.
8. Configure and troubleshoot basic operations of routers in a small routed network using Routing Information Protocol (RIPv1 and RIPv2).
9. Configure and troubleshoot basic operations of routers in a small routed network using Configure Open Shortest Path First (OSPF) protocol (single-area OSPF).
10. Configure and troubleshoot VLANs, Wireless LANs and inter-VLAN routing.
11. Describe the purpose and types of access control lists (ACLs).
12. Prepare, monitor, and troubleshoot access control lists (ACLs) for IPv4 and IPv6.
13. Describe the operations and benefits of Dynamic Host Configuration Protocol (DHCP) and Domain Name System (DNS) for IPv4 and IPv6.
14. Describe the operations and benefits of Network Address Translation (NAT).
15. Configure and troubleshoot Network Address Translation (NAT) operations.
16. Configure and troubleshoot redundancy on a switched network using STP and Ether Channel.
17. Develop critical thinking and problem-solving skills using real equipment and Cisco Packet Tracer.
18. Explain how to support available and reliable networks using dynamic addressing and first-hop redundancy protocols.
19. Configure port security to mitigate MAC address table attacks.

110904 LAN Switching and Wireless/Scaling Networks/Cisco III

This course examines the architectures and design considerations involved in securing, operating, and troubleshooting enterprise networks. Students explore wide area network (WAN) technologies, quality of service (QoS) mechanisms, and secure remote access solutions within modern and emerging technology environments. Instruction includes software-defined networking (SDN), virtualization, and network automation concepts that support scalable and resilient digital infrastructures. Students configure routers and switches for advanced functionality, implement enterprise-level network services, and analyze performance across distributed systems. Emphasis is placed on network scalability, security integration, and structured troubleshooting methodologies. Students apply command-line configuration, automation, and scripting techniques to deploy, secure, and optimize enterprise network infrastructure. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Configure and troubleshoot DHCP and DNS operations for IPv4 and IPv6.
2. Describe the operations and benefits of the Spanning Tree Protocol (STP).
3. Configure and troubleshoot STP operations.
4. Describe the operations and benefits of link aggregation and Cisco VLAN Trunk Protocol (VTP).
5. Configure and troubleshoot VTP, STP, and RSTP.
6. Configure and troubleshoot basic operations of routers in a complex routed network for IPv4 and IPv6.
7. Configure Open Shortest Path First (OSPF) protocol (single-area OSPF and multi- area OSPF).
8. Configure Enhanced Interior Gateway Routing Protocol (EIGRP).
9. Configure and troubleshoot the advanced operation of routers and implement RIP, OSPF, and EIGRP routing protocols for IPv4 and IPv6.
10. Configure and manage Cisco IOS Software licensing and configuration files.
11. Mitigate threats and enhance network security using access control lists and security best practices.
12. Develop critical thinking and problem-solving skills using real equipment and Cisco Packet Tracer.
13. Understand virtualization, SDN, and how APIs and configuration management tools enable network automation.

110906 Network Hardware Installation and Troubleshooting

This course provides students with the knowledge and skills necessary to design, install, configure, and troubleshoot cabling systems and hardware used to support local area networks (LANs). Students examine industry standards for structured cabling, network hardware components, and installation practices within modern and emerging technology environments. Instruction emphasizes safe installation procedures, equipment configuration, signal testing, documentation practices, and systematic troubleshooting methodologies. Students analyze network performance issues related to physical infrastructure and implement corrective actions to ensure reliable connectivity. Students apply configuration tools and diagnostic techniques to test, install, and troubleshoot network hardware and cabling systems. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Design a basic network layout using copper and/or fiber optic cabling systems.
2. Terminate, test, and troubleshoot copper wire systems.
3. Install and configure network interface cards and connection equipment.
4. Use industrial standard testing and certification equipment.

110912 Security Fundamentals

Security Fundamentals introduces foundational computer and network security concepts and methodologies within modern and emerging technology environments. Students examine principles of security, compliance, and operational security, as well as threats and vulnerabilities affecting systems and data. Instruction includes network security, application, data, and host security; access control and identity management; and foundational cryptography concepts. Students analyze risk, evaluate security controls, and apply structured problem-solving techniques to protect digital assets. Students apply configuration, scripting, and analytical techniques to identify vulnerabilities and support the implementation of secure computing practices. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Explain basic security concepts.
2. Identify and explain the appropriate use of security tools to facilitate security.
3. Evaluate current security issues related to computer and network systems.
4. Evaluate and select appropriate incident response procedures, disaster recovery, and risk identification techniques to ensure business continuity.
5. Differentiate various malware and systems security threats against computers and networks.
6. Explain the vulnerabilities and mitigations associated with computers and network devices.
7. Explain the proper use of common tools for carrying out vulnerability assessments.
8. Identify and describe potential application and data vulnerabilities, including buffer overflow, DLL injection, and SQL injection.
9. Explain how host firewalls, malware protection, and updates are important to application and data security.
10. Describe the importance of user accounts and associated permissions.
11. Compare and discuss logical and physical access control security methods.
12. Explain authentication models and identify the components of each model.
13. Summarize and explain general cryptography concepts.
14. Demonstrate public and private key pairs for digital signing and encryption/decryption.

110913 Microsoft Client/Server Configuration

This course covers the installation, configuration, and management of Microsoft Windows client and server operating systems within modern and emerging technology environments. Students examine system architecture, user and group management, resource sharing, security policies, and network services that support enterprise infrastructure. Instruction emphasizes system deployment, configuration management, troubleshooting methodologies, and operational security practices within client/server environments. Students analyze system performance and reliability while implementing administrative controls that support secure and efficient network operations. Students apply configuration tools, command-line utilities, and scripting techniques to deploy, manage, and troubleshoot client and server systems. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Evaluate hardware compatibility for installation.
2. Install and upgrade the Windows client operating system and migrate user data.
3. Manage file systems, partitions, disks and devices.
4. Install, configure, and control access to applications, including desktop applications and Windows Store apps.
5. Configure Client Hyper-V.
6. Configure Transmission Control Protocol/Internet Protocol (TCP/IP) settings and network security settings.
7. Configure Remote Management and remote connections.
8. Configure local and shared access for files and folders.
9. Configure, secure and manage mobile computing.
10. Monitor and optimize operating system performance and resource usage.
11. Implement disaster protection, including backup and file recovery options.
12. Create and manage user accounts and groups.
13. Configure printing and print services.
14. Configure and troubleshoot the boot process and System Recovery options.

110917 Internet Technologies

This course provides students with a study of traditional and emerging Internet technologies, including Internet fundamentals, applications, delivery systems, and client/server computing models. Students examine how Internet architecture supports communication, data exchange, and distributed services within modern digital environments. Instruction emphasizes structured problem solving, data transmission concepts, web-based applications, and foundational programming within Internet-based systems. Students analyze how Internet technologies function, interact, and evolve to support emerging technologies and global connectivity. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Describe the history of the Internet and its impact on government, society, and business.
2. Describe the models used to organize Internet technologies.
3. Explain how the Internet is governed and the standards that are used.
4. Describe the protocols that make the Internet work.
5. Use Internet technologies for data transfer, remote access, information delivery, e-mail, content presentation, and real-time collaboration.
6. Describe how the Internet is used for e-commerce.
7. Describe Internet naming conventions, URLs, and web server file organization.
8. Describe core connectivity issues such as NAT, ISPs, and IP addresses.
9. Create and publish simple web content using basic HTML.
10. Use existing scripting applications and create simple client/server applications to enhance information delivery.

110918 Computer Science Co-op*

Cooperative Education for Computer Science provides supervised worksite experience directly related to the student's identified computer science career pathway. Students must be enrolled in an approved computer science course during the same school year in which the co-op experience is completed. Students participating in cooperative education receive wages in accordance with local, state, and federal minimum wage requirements as outlined in the Work-Based Learning Manual. Through structured workplace learning in modern and emerging technology environments, students apply technical knowledge, professional skills, and industry practices aligned to their pathway. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs within authentic workplace contexts.

Recommended Grade Level: 11 – 12

Recommended Credit: 1

Students will:

1. Demonstrate and practice safe work habits at all times.
2. Gain career awareness and the opportunity to test career choices.
3. Receive work experience related to career interests.
4. Integrate classroom studies with work experience.
5. Receive exposure to facilities and equipment unavailable in a classroom setting.
6. Increase employability potential.

* In section 2 of 705 KAR 4:041, Cooperative education shall meet the minimum requirements established in this section:
(a) Enrolled in a course included within the student's chosen career pathway within the same academic year; or
(b) A career pathway completer by the conclusion of the student's junior year; or
(c) Enrolled in an approved registered youth or pre-apprenticeship program.

110919 Computer Science Internship

Internship for Computer Science provides supervised work-site experience for high school students enrolled in a course aligned to their identified computer science career pathway. Internship experiences combine classroom instruction with field-based learning in professional environments. A student receiving compensation participates in an internship lasting a semester or longer and establishes an employer-employee relationship. Non-paid internships are short-term experiences, typically lasting a semester or less. All internship experiences are conducted in accordance with the Work-Based Learning Manual. Through authentic workplace engagement in modern and emerging technology environments, students apply technical knowledge, professional skills, and industry practices relevant to their pathway. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs within real-world contexts.

Recommended Grade Level: 11 – 12

Recommended Credit: 1

Students will:

1. Demonstrate and practice safe work habits at all times.
2. Gain career awareness and the opportunity to test career choices.
3. Receive work experience related to career interests.
4. Integrate classroom studies with work experience.
5. Receive exposure to facilities and equipment unavailable in a classroom setting.
6. Increase employability potential.

110920 AP Networking

AP Networking is a full-year course that prepares students for entry-level study in computer networking and aligns with first-year collegiate coursework. Students develop problem-solving and communication skills by configuring, securing, and troubleshooting networks of increasing scale. Through hands-on learning and real-world scenarios, students design, implement, and manage networked systems that support communication, collaboration, media streaming, and resilient infrastructure. Instruction emphasizes the use of networking protocols, tools, and technologies to analyze performance and maintain secure, reliable systems. Students explore how networks operate across a variety of environments, including local, enterprise, and emerging technology systems, while developing skills in network architecture, addressing, routing, switching, and security. Learning experiences may include automation and emerging technologies to support network management and optimization. The course also examines the role of networking in modern society, including its impact on communication, access, and infrastructure. Developed in collaboration with postsecondary and industry partners, the course aligns to the NICE Workforce Framework to support career readiness in networking and IT fields. Participation in the Technology Student Association (TSA) or SkillsUSA will greatly enhance instruction.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Identify and describe the components and roles of various network types, including LAN, WAN, WLAN, and PAN.
2. Explain the purpose and structure of the OSI and TCP/IP models and map protocols to appropriate layers.
3. Define and configure IPv4 and IPv6 addressing schemes, including subnetting and CIDR notation.
4. Compare and contrast transmission media (copper, fiber, and wireless) and explain signal transmission methods.
5. Set up and troubleshoot basic network hardware such as switches, routers, firewalls, and access points.
6. Analyze MAC addressing and the role of ARP in network communication.
7. Describe the functions of key networking protocols including TCP, UDP, HTTP, FTP, DNS, and DHCP.
8. Use command-line tools (e.g., ipconfig, ping, traceroute, nslookup, netstat) to analyze and troubleshoot networks.
9. Configure static and dynamic IP addressing and demonstrate the ability to set up DHCP services.
10. Design a basic small office/home office (SOHO) network with considerations for scalability and efficiency.
11. Create and interpret network diagrams using standard notation and software tools.
12. Implement basic security practices including password policies, firewalls, and wireless encryption standards (e.g., WPA3).
13. Describe authentication, authorization, and accounting (AAA) principles and their role in network security.
14. Explain how NAT, PAT, and port forwarding work in managing IP address spaces and enabling external communication.
15. Interpret packet captures using tools such as Wireshark to analyze and troubleshoot

network traffic.
16. Recognize common network threats (e.g., DDoS, spoofing, phishing) and explain mitigation techniques.
17. Understand and apply wireless networking standards (e.g., 802.11a/b/g/n/ac/ax) and security protocols.
18. Describe concepts of redundancy and fault tolerance in network design, including load balancing and failover systems.

The student performance expectations currently listed for this course are provisional and are being used for temporary instructional guidance purposes only. These expectations are based on previously available networking standards and legacy course outcomes while the official AP Networking framework is under development by the College Board. Teachers should refer to the most current course framework, course audit requirements, and instructional guidance available through AP Central at collegeboard.org. Upon release of the finalized framework, these expectations may be revised to ensure alignment.

111001 Computers, Networks, and Databases

This project-based learning course engages students in the design and development of systems that acquire, store, manage, and communicate data across computing and networked environments. Students apply a structured design process to create data-driven solutions relevant to a variety of career fields within modern and emerging technology environments. Instruction emphasizes computational thinking, systems architecture, data representation, and database design principles. Students collaborate in teams to analyze problems, design integrated systems, think critically, apply creativity, and communicate technical solutions to peers and business partners. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Use the technical design process to design, build, and test prototypes.
2. Use the terminology of the field.
3. Use data and informatics tools to make decisions and solve problems.
4. Apply project management principles.
5. Use appropriate and effective research skills.
6. Demonstrate proficiency in word processing, spreadsheets, databases, and presentation software.
7. Communicate information, including descriptive statistics, to various stakeholder groups.

111003 Databases in the Cloud

This project-based learning course engages students in solving complex business and industry challenges through the design and implementation of cloud-based data systems. Students explore cloud storage architectures, web technologies, database applications, information security, and introductory programming within modern and emerging technology environments. Instruction emphasizes data acquisition, storage, transformation, and communication using scalable cloud infrastructures. Students apply structured problem-solving and systems design principles to build, test, analyze, and improve data-driven solutions. Real-world challenges require higher levels of research, collaboration, critical thinking, and evaluation of data integrity, performance, and security. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Use the terminology of the field.
2. Research data science technical texts, journal articles, and other related documents in developing a plan.
3. Use the five-step software/system life cycle (design, build, test, implement, and evolve).
4. Use data science concepts to solve problems.
5. Use data and data science tools to make decisions and solve problems.
6. Apply project management principles.
7. Gain information on how the American computer industry works.
8. Use appropriate and effective research skills.
9. Use best practices to design and implement research studies.
10. Use the scientific method to design investigations.
11. Demonstrate proficiency in word processing, spreadsheets/databases, and presentation software.
12. Communicate information, including descriptive statistics, to various audiences.

111004 Data Visualization

This project-based course engages students in analyzing, interpreting, and communicating data through visual representations that support decision-making. Students explore principles of data ethics, privacy, and responsible data use while designing visualization solutions for authentic clients within modern and emerging technology environments. Student teams collaborate with a business partner to identify data needs, develop a project proposal with evaluation criteria, and design, build, test, and refine interactive data visualizations. Instruction emphasizes data representation, visual design principles, user-centered communication, and critical evaluation of how data insights are presented and interpreted. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs used to analyze and visualize data. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Use the terminology of the field.
2. Research data science technical texts, journal articles, and other related documents in developing a plan.
3. Use the five-step software/system life cycle (design, build, test, implement, and evolve).
4. Use data science concepts to solve problems.
5. Use data and data science tools to make decisions and solve problems.
6. Apply project management principles.
7. Gain information on how the American computer industry works.
8. Use appropriate and effective research skills.
9. Use best practices to design and implement research studies.
10. Use the scientific method to design investigations.
11. Demonstrate proficiency in word processing, spreadsheets, databases, and presentation software.
12. Communicate information, including descriptive statistics, to various audiences.

113601 Introduction to Digital Game Graphics

This course introduces students to the foundational principles of digital game graphics and interactive design within industry-aligned game development environments. Students create game assets using animation techniques and introductory 3D design software while exploring how visual elements support gameplay and user experience. Instruction emphasizes creative problem solving, visual storytelling, asset integration, and the role of programming in bringing interactive elements to life within modern and emerging technology platforms. Students apply design processes to develop, test, and refine game graphics and interactive components that align with industry expectations. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs used within game development contexts. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Identify the target audience of a game.
2. Explain the impact of "feature creep" on production.
3. Explain the interdependence of team members between artistic, technical and production disciplines.
4. Explain the purpose of prototyping.
5. Outline in detail the process of developing a game from concept to delivery and support.
6. Describe each step of the production process.
7. Explain how the project is going to be managed according to a milestone plan.
8. Explain the various types of collaboration tools.
9. Utilize the production pipeline in the development of a game.
10. Explain the value of version control.
11. Explain the purpose of vertical slice.
12. Demonstrate version control for example, Node Version Manager (NVM).
13. Demonstrate good quality assurance practices.
14. Conceptualize and illustrate original game characters and assets.
15. Utilize illustration to create assets.
16. Establish a standard for world scale.
17. Create a storyboard for planning animation.
18. Change an object's state or position over time.
19. Establish an object's relative speed.
20. Simulate a naturally occurring or mechanical cycle such as walking.
21. Apply animation to game assets.
22. Describe the role of typography.
23. Evaluate the use of layout and composition.
24. Explain color theory.
25. Describe the principles of animation.
26. Describe the role of perspective.
27. Demonstrate 1- and 2-point perspectives.
28. Draw a proportionally correct figure.
29. Describe the characteristics and purposes of 2D, 2.5D, and 3D art.
30. Recognize the importance of and implement continuity of art style.

31. Describe environments within a game.
32. Compare the process of creating an interior vs. exterior environment.
33. Identify components in an environment.
34. Generate terrains for a specific environment.
35. Create hard surface assets.
36. Create an environment.
37. Develop organics for a specific environment.
38. Describe archetypes of characters.
39. Explain character personalities and stereotypes.
40. Compart and contrast methods to design characters.
41. Describe the character's evolution throughout the game.
42. Examine the importance of non-player characters (NPC).
43. Construct character(s) for a game.
44. Differentiate between syntax and semantics.
45. Incorporate primitive data types.
46. Utilize arrays to store a list of primitive data types.
47. Demonstrate input from different sources.
48. Construct and register a callback function.
49. Compare and contrast constants and variables.
50. Select and implement conditional control.
51. Implement functions.
52. Select and implement iterations (loops, recursion).
53. Recognize and implement sequential control.
54. Test and debug programs.
55. Design and implement user-defined data types.
56. Demonstrate output to different destinations.
57. Practice object-oriented programming (OPP).
58. Identify expected input and output.
59. Utilize basic steps in algorithmic problem-solving.
60. Discuss top-down versus bottom-up development.
61. Generate test cases and expected results.
62. Apply simple data structures.
63. Explain how algorithms are used to produce artificial intelligence (AI).

113602 Advanced Game Development and Publishing

This course focuses on the advanced development, production, and publishing of interactive digital games using industry-aligned game engines. Students design and implement complex gameplay systems through coding, 3D character and object development, animation, and asset integration within modern and emerging technology platforms. Instruction emphasizes advanced problem solving, project management, optimization, testing, and iterative improvement of game systems. Students create industry-ready products, participate in Game Jams to practice collaboration and deadline-driven development, and prepare projects for deployment and distribution. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs within game development contexts. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Compare and contrast licensed vs. proprietary game engines.
2. Debate the strengths and weaknesses of various game engines.
3. Discuss the impact of a game engine on the development of a game.
4. Explain how game engines work.
5. Explain character advancement in relation to storyline and gameplay.
6. Define the size of the player environment.
7. Explain the locations and purpose of non-player characters (NPC).
8. Specify boundaries and borders of the levels within the game.
9. Justify placement of triggers and scripted events.
10. Develop a game with multiple levels.
11. Research types of GUI.
12. Recognize and implement required feedback for the GUI.
13. Create a flowchart that maps the GUI's functionality.
14. Design and implement a GUI using wireframes.
15. Create a victory condition.
16. Assemble immersive elements into a game.
17. Establish reward systems and in-game economies.
18. Apply game mechanics to the game world.
19. Balance and test game mechanics.
20. Integrate different types of audio, including sound effects, ambient background, dialog, and score.
21. Practice creating sound loops.
22. Determine acceptable media files for game development, such as sound, graphics, and video.
23. Import appropriate media for a game.
24. Incorporate feedback sounds.
25. Compare and contrast the benefits of various platforms and their target markets.
26. Evaluate the need for flexibility and scalability when developing for a PC.
27. Explore development tools specific to various consoles.
28. Research procedures to deliver a game to mobile markets.
29. Pitch a project and defend why it is entertaining.
30. Explain the role of social media in marketing.

31. Describe crowdsourcing and crowdfunding.
32. Explain the merchandising and branding behind video games.
33. Analyze successful trailers.
34. Explain the concepts of localization and its impact on design.
35. Describe various pay models, such as free-to-play, pay-to-play, singer-user license, and freemium.
36. Describe the integration of social components in a game.
37. Explain the role of social media in the gaming community.
38. Describe professional events in digital gaming.
39. Summarize characteristics of cloud gaming.
40. Evaluate the advances of multi-player gaming.
41. Discuss trends in input devices.
42. Examine current trends in output devices and displays.
43. Explore advances in peripheral devices.

113603 Advanced 3D Game Development

This course emphasizes the creation of advanced 3D graphics and interactive simulations using state-of-the-art industry software and development tools. Students gain a thorough understanding of techniques used to design and implement complex 3D games and immersive environments within modern and emerging technology platforms. Instruction includes 2D and 3D graphics, animation, character development, texturing, rigging, scripting, physics integration, and game system configuration. Students apply advanced design principles and technical workflows to develop optimized, interactive 3D experiences aligned with industry expectations. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs within advanced game development contexts. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Compare and contrast modeling methodologies (for example, polygons, NURBS, splines).
2. Explain the applications of low polygon and high polygon construction.
3. Construct and manipulate polygonal objects.
4. Applying texturing/surfacing/shading to models and normal mapping.
5. Identify UVW mapping coordinates.
6. Explain how lighting and shading effect form and surface.
7. Implement basic lighting concepts for ambient and artificial light.
8. Describe the difference between forward and inverse kinematics.
9. Examine the process of particle creation and its application to game design.
10. Create a parent/child hierarchy.
11. Create a joint/bone chain.
12. Apply and adjust weight maps.
13. Create atmospheric effects.
14. Demonstrate the use of constraints to animate objects.
15. Apply various animation techniques (for example, pose-to-pose, straight ahead).
16. Adjust the dynamic properties (for example, gravity and wind speed).
17. Simulate rigid body dynamics (shattering wall, breaking glass).
18. Utilize cinematography in animation.
19. Describe the process of motion capture for animation.

113604 Digital 3D Graphics and Special Effects II

This course focuses on the development of advanced 3D graphics and special effects within industry-aligned game engines and digital production environments. Students create interactive projects using code, 3D characters, objects, animation, and visual effects workflows designed to meet professional industry standards. Instruction emphasizes advanced texturing, lighting, rendering techniques, and visual composition to enhance depth perception, realism, and immersive experiences within modern and emerging technology platforms. Students apply scripting and technical integration processes to refine visual systems and optimize performance. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs within advanced 3D and visual effects contexts. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Demonstrate an understanding of how to add textures to objects.
2. Use appropriate types of lighting techniques to design.
3. Demonstrate adding depth using different types of shadowing techniques.
4. Create custom connections and color utilities for innovative designs.
5. Use indirect and direct illumination to designs.

113605 Game Design and Development Principles

This course provides an introduction to game design and the gaming industry through the exploration of storytelling, gaming history, game critique, current trends, and industry software. Students examine core design principles and begin developing original game concepts, including storylines and gameplay mechanics that can be expanded in advanced coursework. Instruction emphasizes creative problem solving, visual design, user experience, and foundational development processes. Students explore 2D game development tools and image manipulation techniques to support interactive design while analyzing how games function within modern and emerging technology platforms. Career exploration activities increase awareness of industry roles and postsecondary opportunities in game design and development. Students are actively engaged with code to develop computational thinking and problem solving skills by developing, interpreting, evaluating, and revising programs within introductory game development contexts. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 9 – 12

Recommended Credit: 1

Students will:

1. Explain the history of computing technologies that impact the game development industry.
2. Explore non-digital games.
3. Research the evolution of video games.
4. Describe the different game genres.
5. Evaluate the contributions of individual game designers and developers.
6. Explore careers as a game artist and sound designer.
7. Describe the role of a game designer.
8. Explore careers as a game developer.
9. Describe career pathways in quality assurance/testing.
10. Explain the role of the producer.
11. Explain the career path of an independent developer.
12. Research salary structures in the industry.
13. Define common terminology and its acronyms.
14. Identify the tools to develop a game (engine, application program interface [API], digital content creation tools, editors).
15. Communicate both in writing and verbally using appropriate industry terminology.
16. Compare and contrast the entertainment software rating boards (ESRB) ratings for games.
17. Explain the principles of visual design.
18. Explain the elements of design.
19. Analyze artwork/designs for specific design theories.
20. Explore the components of game structure.
21. Analyze the essentials of storytelling.
22. Write an outline of a nonlinear story.
23. Create rules for a game.
24. Compare conflict and outcomes.
25. Develop objectives and outcomes for a game.
26. Explain the importance of usability and how it impacts user experience.
27. Explain in-game economies, motivators, and reward systems.
28. Research various styles of game documentation.

29. Develop a technical design document (TDD).
30. Describe components of a game design document (GDD).
31. Produce a game design document.
32. Produce a game pitch document.
33. Present game documentation.
34. Compare and contrast categories of game mechanics.
35. Research victory condition mechanics of a game.
36. Discuss the relationship between game mechanics and game complexity and interactions.
37. Incorporate game mechanics into a game.
38. Explain basic logic statements such as if/then and cause/effect.
39. Describe the uses of Boolean operators and the symbols associated with them.
40. Generate truth tables for game events.
41. Examine different number systems, including binary, decimal, and hexadecimal.
42. Demonstrate proper use of the order of operations.
43. Convert mathematical formulas into code.
44. Explain when to apply mathematical concepts common to game coding.
45. Use logical thinking to create a diagram of code execution.
46. Research laws that govern intellectual property in diverse forms.
47. Evaluate Creative Commons and open-source licensure.
48. Cite the boundaries of third-party work.
49. Explain copyright, trademarks, and other intellectual property protection.
50. Explain the invasion of privacy in the use of technology.
51. Model acceptable security practices.
52. Explore the issues of piracy and digital rights management (DRM).
53. Analyze your digital footprint.
54. Discuss social responsibility and issues concerning gaming.
55. Model legal and ethical use of information.
56. Identify key elements of non-disclosure agreements (NDA) and contracts.
57. Summarize the behavior of an algorithm.
58. Determine if a given algorithm successfully solves a stated problem.

210241 Introduction to Geographical Information Systems (GIS)

This introductory course provides foundational theories and concepts of geographical information systems (GIS), including spatial data collection, data types, GPS technologies, and mapping principles. Students explore core GIS capabilities and industry-specific software applications to understand how geographic data is captured, stored, analyzed, and visualized within modern and emerging technology environments. Instruction emphasizes spatial data representation, analysis, and problem solving to support data-driven decision-making across multiple career fields. Students apply structured design processes to interpret geographic datasets and create meaningful visual representations. Students are actively engaged with code to develop computational thinking and problem solving skills by writing queries and scripts to analyze, manage, and visualize spatial data within GIS platforms. Participation in the Kentucky Technology Student Association (TSA) or SkillsUSA is encouraged to enhance applied learning experiences.

Recommended Grade Level: 10 – 12

Recommended Credit: 1

Students will:

1. Explain what GIS is and the practical applications used in this field.
2. Use GIS software to edit basic spatial and attribute data.
3. Use GIS software to create and use basic geo-databases.
4. Explain basic topics of GIS, such as spatial data and attribute data management.
5. Explain the human and organizational issues.
6. Explain the differences between vector and raster data.
7. Use GIS software to create basic query features.
8. Use GIS software to build basic graphs, reports and personal systems.
9. Solve route problems from sets of interconnected lines using a network analysis program.
10. Combine layers of GIS data and locate areas of special concern using a spatial analysis program.
11. Create 3-dimensional representations of landscapes and other surfaces using satellite and aerial photographic images using a 3D analysis program.