

Multi-User Game & Simulation Programming (2026)

Identify and describe basic network terminology and network security. The student will be able to: 52.0

- 1 Define networking and describe the purpose of a network. 52.01
- 2 Identify the purposes and interrelationships among the major components of networks (e.g., servers, clients, transmission media, network operating system, network boards). 52.02
- 3 Describe the various types of network topologies. 52.03
- 4 Describe the various types of game protocols 52.04
- 5 Demonstrate knowledge of general security concepts. 52.05
- 6 Develop an awareness of communication security concepts. 52.06
- 7 Develop an awareness of network infrastructure security. 52.07
- 8 Describe the various types of multiplayer game architectures. 52.08
- 9 Identify networking and server design requirements for multi-player games. 52.09
- 10 List and describe performance metrics for networked games. 52.10

Game configuration. The student will be able to: 53.0

- 1 Create a window to run a game. 53.01
- 2 Describe and use appropriate game libraries to run a windowed game. 53.02
- 3 Use reference materials such as on-line help, vendor bulletin boards, tutorials, and manuals available. 53.03
- 4 Troubleshoot problems with computer hardware based on different graphic modes of the game. 53.04
- 5 Describe ethical issues and problems associated with computer games. 53.05
- 6 Read and comprehend technical and non-technical reading assignments related to course content including trade journals, books, magazines and electronic sources. 53.06

-
- 7 Respond to and utilize information derived from multiple sources (e.g., written documents, instructions, e-mail, voice mail) to solve business problems and complete business tasks. 53.07**
 - 8 Explore, design, implement, and evaluate organizational structures and cultures for managing project teams. 53.08**
 - 9 Identify characteristics of operating systems and graphics pipeline. 53.09**
 - 10 Distinguish among integer and floating-point bounding box collision calculations. 53.10**
 - 11 Illustrate various configurations of software libraries. 53.11**
-

Test programs. The student will be able to: 54.0

- 1 Develop data for use in program testing. 54.01**
 - 2 Perform debugging activities. 54.02**
 - 3 Distinguish among the different types of program and design errors. 54.03**
 - 4 Evaluate program test results. 54.04**
 - 5 Execute programs and subroutines as they relate to the total application. 54.05**
 - 6 Use trace routines of compilers to assist in program debugging. 54.06**
 - 7 Compile and run programs. 54.07**
-

Plan program design. The student will be able to: 55.0

- 1 Formulate a plan to determine program specifications individually or in groups. 55.01**
 - 2 Use a graphical representation or pseudo code to represent the structure in a program or subroutine. 55.02**
 - 3 Design programs to solve problems using problem-solving strategies. 55.03**
 - 4 Prepare proper input/output layout specifications. 55.04**
 - 5 Examine existing utility programs and subroutines for use with other programs. 55.05**
 - 6 Manually trace the execution of programs and verify that programs follow the logic of their design as documented. 55.06**
-

Create and maintain documentation. The student will be able to: 56.0

- 1 Write documentation to assist operators and end-users. 56.01**
 - 2 Follow established documentation standards. 56.02**
 - 3 Update existing documentation to reflect program changes. 56.03**
-

Code programs. The student will be able to: 57.0

- 1 Utilize reference manuals.** 57.01

- 2 Write programs according to recognized programming standards.** 57.02

- 3 Write internal documentation statements as needed in the program source code.** 57.03

- 4 Code programs in high-level languages for gaming and simulation applications.** 57.04

- 5 Write code that accesses sequential, indexed sequential, random, and direct files.** 57.05

- 6 Code programs using logical statements (e.g., if-then-else, do...while).** 57.06

- 7 Enter and modify source code using a program language editor.** 57.07

- 8 Code routines within programs that validate input data.** 57.08

- 9 Use the rounding function in calculations within programs.** 57.09

- 10 Write programs that display text.** 57.10

- 11 Demonstrate proficiency in drawing lines using graphic primitive functions.** 57.11

- 12 Demonstrate proficiency in drawing rectangles using graphic primitive functions.** 57.12

- 13 Demonstrate proficiency in drawing circles using graphic primitive functions.** 57.13

- 14 Demonstrate proficiency in drawing ellipses using graphic primitive functions.** 57.14

- 15 Demonstrate proficiency in drawing polygons using graphic primitive functions.** 57.15

- 16 Write programs that use composite graphic objects.** 57.16

- 17 Write programs that load a bitmap for background.** 57.17

- 18 Write programs that use a sprite handler.** 57.18

- 19 Write programs that use animation.** 57.19

- 20 Write programs that use scrolling.** 57.20

- 21 Write programs that use transparency.** 57.21

Demonstrate an understanding of operating systems, environments, and platforms. The student will be able to: 58.0

- 1 Identify various types of operating systems/environments for different computer hardware platforms.** 58.01
- 2 Assess and analyze the functions of different operating systems.** 58.02
- 3 Distinguish between different types of computer hardware platforms.** 58.03

Implement enhanced program structures. The student will be able to: 59.0

- 1 Write programs that include tables or arrays and routines for data entry and lookup.** 59.01
- 2 Write routines to sort arrays.** 59.02
- 3 Write programs that sort records in files.** 59.03
- 4 Write programs to process transactions.** 59.04
- 5 Write programs that use iteration.** 59.05
- 6 Write programs that read and write sequential files.** 59.06
- 7 Write programs that read and write random files.** 59.07

Implement multimedia programming. The student will be able to: 60.0

- 1 Demonstrate proficiency in creating multiple composite objects.** 60.01
- 2 Demonstrate proficiency in moving composite graphics objects.** 60.02
- 3 Demonstrate proficiency in rotating composite graphics objects by hand.** 60.03
- 4 Distinguish between flock and flee artificial intelligence algorithms.** 60.04
- 5 Write programs that use blitting.** 60.05
- 6 Simulate circular game board.** 60.06
- 7 Demonstrate proficiency in creating a firing simulation.** 60.07
- 8 Identify the basic constructs used in bounding box collision algorithm.** 60.08
- 9 Identify the basic constructs used in truer bounding box collision.** 60.09
- 10 Demonstrate proficiency in creating a creating a bouncing simulation.** 60.10
- 11 Simulate pattern-based movement.** 60.11
- 12 Simulate multiple sprites movement.** 60.12
- 13 Identify the basic constructs used in keyboard input.** 60.13
- 14 Identify the basic constructs used in mouse input.** 60.14

15 Identify the basic constructs used in double buffering. 60.15

Develop an understanding of programming techniques and concepts. The student will be able to: 61.0

1 Identify the basic constructs used in structured programming. 61.01

2 Distinguish between top-down and bottom-up design. 61.02

3 Distinguish between iteration and recursion. 61.03

4 Evaluate Boolean expressions. 61.04

5 Distinguish between interpreters and compilers. 61.05