Name: _____ **Business Computer Programming**

Directions:
Evaluate the student by checking the appropriate number or letter to indicate the degree of competency. The rating for each task should reflect **employability readiness** rather than the grades given in class.

Rating Scale:
- **3 Mastered** – can work independently with no supervision
- **2 Requires Supervision** – can perform job completely with limited supervision
- **1 Not Mastered** – requires instruction and close supervision
- **N No Exposure** – no experience or knowledge in this area

3	2	1	N	A. Explore Computer Concepts	Notes:
				1. Trace the development of computers and their impact on society	
				2. Describe the categories and evolution of programming languages	
				3. Explain the functions of computer hardware and architecture	
				4. Demonstrate an understanding of computer theory (e.g., bits, bytes, binary logic, memory, and storage)	
				5. Compare computer operating systems (e.g., DOS, Windows, and Unix)	
				6. Discuss legal/ethical issues related to computers	
				7. Identify the application environment/interface for the specific language being covered (e.g., Windows, Macintosh, or DOS Based)	
				8. Explain the concept of security and its relationship to programming	
				9. Manage the operating system on the workstation	
				10. Explain the difference between a mainframe, midframe, server, and personal computer	
				Other:	

3	2	1	N	B. Apply Logical Problem-Solving Skills	Notes:
				1. Analyze a problem	
				2. Determine the steps needed to solve a problem	
				3. Create a method to solve a problem	
				4. Illustrate the problem solution using a flowchart or pseudocode	
				Other:	

3	2	1	N	C. Describe the Software Development Life Cycle	Notes:
				1. Explain how requirements for a new program are gathered	
				2. Explain how to analyze the requirements for a new program	
				3. Explain how to create a flowchart or pseudocode for a new program	
				4. Explain how to use a flowchart or pseudocode in coding the modules of a new program	

				5. Explain how to integrate the modules of a new program	
				6. Explain how a new program is authorized/accepted	
				7. Explain how to maintain a program	
				Other:	

3	2	1	N	D. Develop Program Applications	Notes:
				1. Use correct syntax of a given programming language	
				2. Create a program using internal documentation	
				3. Create a program using variables and constants	
				4. Create a program using counters and accumulators	
				5. Create a program using arithmetic operations and functions	
				6. Create a program using a conditional statement	
				7. Create a program using a loop instruction	
				8. Create a program that requires user input	
				9. Create a program that includes input validation	
				10. Create a program to open, write, and read from a data file	
				11. Create a program to produce a report	
				12. Create a modular program using one or more subroutines	
				13. Create a program using one- and two-dimensional arrays	
				14. Create a program using a sort routine	
				15. Create a program with a standard Windows graphic user interface (GUI) with objects and menus	
				16. Create a program with a custom GUI	
				17. Create an object-oriented program by creating objects and classes	
				18. Create a program to display graphics	
				19. Create a program to animate graphic objects	
				20. Create a program using multimedia	
				21. Create a program and supporting external documentation	
				22. Modify an existing program	
				23. Create a program in collaboration with a team	
				Other:	

3	2	1	N	E. Explore Additional Programming Concepts	Notes:
				1. Describe steps involved in troubleshooting and debugging	
				2. Discuss considerations in programming for efficiency (e.g., computer time, programmer time, etc.)	
				3. Discuss how to create a user-friendly program	
				4. Describe event-driven programming	
				5. Describe error catching/handling	
				6. Compare object-oriented programming with structured programming	
				7. Describe how the Internet uses programming	
				8. Explain uses of scripting languages	
				9. Discuss handicap accessibility considerations in programming	
				Other:	

3	2	1	N	F. Apply Database Concepts	Notes:
				1. Create file structures	
				2. Describe database structures (e.g., fields, records, files, and tables)	
				3. Create a database file with one or more tables for manipulation by program code	
				4. Create a database file with one or more tables via text editor, database software, and/or source code	
				5. Write code to append, delete, and update a table or a file	
				6. Write code to integrate a database with another application	
				7. Create a relational database application	
				8. Write code to search, sort, and query a database	
				Other:	

3	2	1	N	G. Prepare for Employment	Notes:
				1. Demonstrate working as a team	
				2. Demonstrate analytical skills	
				3. Search the Internet and other places to locate career-planning information and job opportunities related to programming	
				4. Identify careers in the information technology field	
				5. Demonstrate communication skills	

				6. Demonstrate logical thinking	
				7. Demonstrate interpersonal skills	
				8. Explore compatibility for programming	
				Other:	