UML Milestone Rubric – W23 – Barcomb

RUBRIC	100%	> 90%	> 80%	> 70%	> 50%	0
UML Draft	All formatting conventions are correctly followed. Depicted class relationships are justified by the given specifications. All relationships are supported by the necessary data members and all correct cardinalities are included. No missing default values, constructors, getters or setters. Any association classes are shown in the analysis phase. Design shows correct use of object-oriented principles - each class has a clear, distinct purpose (encapsulation). The following components are included: A) inheritance and/or interface, B) aggregation and/or composition. The design makes appropriate use of final, static, and final static fields. Public methods are not assumed to receive correct imput and throw exceptions accordingly. Method names indicate all expected behavior.	Minor formatting errors. 2 or fewer class relationships do not align with given specifications. Minimal cardinality errors. No missing default values, constructors, getters or setters. Any association classes are shown in the analysis phase. Design shows correct use of object-oriented principles - each class has a clear, distinct purpose (encapsulation). The following components are included: A) inheritance and/or interface, B) aggregation and/or composition. The design makes appropriate use of final, static, and final static fields. Public methods are not assumed to receive correct imput and throw exceptions accordingly. Method names indicate all expected behavior.	Minor formatting errors. 4 or fewer class relationships do not align with given specifications. Several cardinality errors. Some missing or incorrect default values, constructors, getters or setters. Object-oriented design is attempted, but unclear. The design makes appropriate use of final, static, and final static fields. Some, but not all, exceptions are included.	Major formatting errors or difficult to read. Many class relationships do not align with the given specifications. Several cardinality errors. Some missing or incorrect default values, constructors, getters or setters. Relationships use linear functionality.	Major formatting errors and difficult to read. The majority of class relationships cannot be justified by the given specifications, but attempt has been made to link objects. Several cardinality errors. Multiple missing elements.	Fails to meet minimum specs.

Testing Milestone Rubric – W23 – Barcomb

RUBRIC	100%	> 90%	> 80%	> 70%	> 50%	0
Test Design	Tests are supported by code modelled in UML design. All test cases are clearly documented. All major functionality is tested and fail messages are descriptive. Boundary cases and exception handling are tested thoroughly. Tests are written with the expectation that methods have discrete functionality and are free of side effects. Exceptions are tested.	Tests are mostly supported by code modelled in UML design. At least 90% of test cases are clearly documented. At least 90% of major functionality is tested and fail messages are descriptive. Multiple boundary cases and exception handling are tested thoroughly. Tests are written with the expectation that methods have discrete functionality and are free of side effects. Exceptions are tested.	Tests are somewhat supported by code modelled in UML design. Test cases include basic comments. At least 75% of major functionality is tested and fail messages are descriptive. Multiple boundary cases and exception handling are tested. Tests are written with the expectation that methods are free of side effects.	Tests are somewhat supported by code modelled in UML design. Test cases are labelled. Base functionality is tested and fail messages are included. A few boundary cases and exception handling are included.	Tests do not align accurately with UML model. Test cases are not documented. Some functionality is tested. Boundary cases and exception handling are missing from tests.	Fails to meet minimum specs.

Final Package Rubric – W23 – Barcomb

RUBRIC	100%	> 90%	> 80%	> 70%	> 50%	0
UML Diagram (15%)	All formatting conventions are correctly followed. Depicted class relationships match the code. All relationships are supported by the necessary data members and all correct cardinalities are included. No missing default values, data members, or methods. Any association classes are shown in the analysis phase. Design shows correct use of object-oriented principles - each class has a clear, distinct purpose (encapsulation). The following components are included: A) inheritance and/or interface, B) aggregation and/or composition. The design makes appropriate use of final, static, and final static fields. Public methods are not assumed to receive correct imput and throw exceptions accordingly. Method names indicate all expected behavior.	Minor formatting errors. 2 or less class relationships are incorrect or missing supporting data members. 2 or less cardinality errors. No missing default values, data members, or methods. Any association classes are shown in the analysis phase. Design shows correct use of object-oriented principles - each class has a clear, distinct purpose (encapsulation). The following components are included: A) inheritance and/or interface, B) aggregation and/or composition. The design makes appropriate use of final, static, and final static fields. Public methods are not assumed to receive correct imput and throw exceptions accordingly. Method names indicate all expected behavior.	Minor formatting errors. 4 or less class relationships are incorrect or missing supporting data members. 4 or less cardinality errors. 3 or less default values, data members, or methods are missing or incorrect. Design shows correct use of object-oriented principles - each class has a clear, distinct purpose (encapsulation). The design makes appropriate use of final, static, and final static fields. Public methods are not assumed to receive correct imput and throw exceptions accordingly. Method names indicate all expected behavior.	Major formatting errors or difficult to read. 6 or less class relationships are incorrect or missing supporting data members. 6 or less cardinality errors. 5 or less default values, data members, or methods are missing or incorrect. The design makes appropriate use of final, static, and final static fields.	Major formatting errors and difficult to read. More than 6 class relationships are incorrect or missing data members. Several cardinality errors. More than 6 default values, data members, or methods are missing or incorrect.	Fails to meet minimum specs.

RUBRIC	100%	> 90%	> 80%	> 70%	> 50%	0
Code Implementation (40%)	Application prompts for user interaction through a GUI. Most efficient food package combination is calculated and output to the user through the GUI. An order form .txt file is generated that includes all the required information and is formatted neatly. The database inventory is updated to reflect items that are no longer available. The program does not exit prematurely, and users are given meaningful error messages, and program is flexible regarding input accepted.	Application prompts for user input through a GUI. Most efficient food package combination is calculated and output to the user through the GUI. An order form .txt file is generated that includes all the required information. The database inventory is updated to reflect items that are no longer available. The program does not exit prematurely, and users are given meaningful error messages, and program is flexible regarding input accepted.	User request is input through a GUI. Most efficient food package combination is calculated and output to the user through the terminal. An order form .txt file is generated that includes some of the required information. The database inventory is updated to reflect items that are no longer available. The program does not exit prematurely.	User request is input through the terminal. Most efficient food package combination is calculated and output to the user through the terminal. An order form .txt file is generated that includes some of the required information.	User request is input through the terminal. Most efficient food package combination is calculated and output to the user through the terminal.	Fails to meet minimum specs.
Documentation (10%)	All active team members' names are included, as well as the code version. All classes and methods include meaningful descriptions about the use of the code, as well as comments throughout to explain the functionality. All formatting and naming conventions follow the ENSF 409 guidelines.	At least 90% of classes and methods include meaningful descriptions about the use of the code, as well as comments throughout to explain the functionality. Less than 5 formatting or naming convention errors.	At least 75% of classes and methods include meaningful descriptions about the use of the code, as well as comments throughout to explain the functionality. Less than 10 formatting or naming convention errors.	At least 50% of classes and methods include meaningful descriptions about the use of the code, as well as a few comments to explain functionality. Less than 15 formatting or naming convention errors.	At least 50% of classes and methods include some description about the code. Frequent formatting or naming convention errors.	Fails to meet minimum specs.

RUBRIC	100%	> 90%	> 80%	> 70%	> 50%	0
Unit Testing (25%)	Tests can be compiled and run from the command line. All test cases are clearly documented. All major functionality is tested and fail messages are descriptive. Boundary cases and exception handling are tested thoroughly.	Tests can be compiled and run from the command line. At least 90% of test cases are clearly documented. At least 90% of major functionality is tested and fail messages are descriptive. Multiple boundary cases and exception handling are tested thoroughly.	Tests can be compiled and run from the command line. Test cases include basic comments. At least 75% of major functionality is tested and fail messages are descriptive. Multiple boundary cases and exception handling are tested.	Tests can be compiled and run from the command line. Test cases are labelled. Base functionality is tested and fail messages are included. A few boundary cases and exception handling are included.	Tests can be compiled and run from the command line. Test cases are not documented. Some functionality is tested. Boundary cases and exception handling are missing from tests.	Fails to meet minimum specs.
Video Demonstration (10%)	Demonstration clearly explains how the solution meets the requirements including the user input/output. The database is shown to be updated and the output form is presented. Audio is clear and audible. All active team members participate in the demonstration and duration is less than 5 minutes.	Demonstration shows all required functionality including the user input/output. The database is shown to be updated and the output form is presented. Audio is clear and audible. All active team members are present and duration is less than 6 minutes.	Demonstration shows all achieved functionality including the user input/output. The output form is presented. Audio is clear and audible. All active team members are present and duration is less than 7 minutes.	Demonstration shows acheived functionality including the user input/output. Audio is clear and audible. Duration is less than 7 minutes.	Demonstration shows acheived functionality. Audio is clear and audible. Duration is over 7 minutes.	Fails to meet minimum specs.