|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | | |  |  | | --- | --- | | |  | | --- | | **HONORS COURSE CONTRACT** | | | | |
| Top of Form   |  |  |  | | --- | --- | --- | | |  |  | | --- | --- | | https://wf.auburn.edu/wfprod/images/spacer.gif | **1. Describe the Honors component of the course, discussing the nature of the project, how this project constituites "Honors," and the nature of the "consolidating experience". Be as specific as possible in describing what you will do, including references to specific readings and/or methodologies where appropriate, lengths of required papers or reports, number and nature of presentations, etc. Research papers are typically 8-10 double spaced pages but certain other projects may include a shorter report.** | |   Bottom of Form |

The student will develop a simple cryptographic tool.

The goals of this extra project are given below:

1. To learn two cryptographic algorithms.
2. To develop a simple cryptographic tool.
3. To perform separate compilation.
4. To develop a command-line application.

**1. Analysis (1 artifact):** Prepare use cases. Remember, these use cases describe how the user interacts with the cryptographic tool via command lines (what does the tool do, what the tool does in response, etc.). Your use cases should have enough basic details such that someone unfamiliar with the cryptographic tool can have an understanding of what is happening in the tool after a command is issued. They should not include internal technical details that the user is not (and should not be) aware of. Check out the following link for a use case example: <http://www.eng.auburn.edu/~xqin/courses/comp2710/useCases.ppt>

**Note**: You have to follow the format of this use case example.

**2.** **Design (2 artifacts)**:

a. Create a Class Diagram (as in Lab 2). The diagram is not optional. Be sure to include:

1) The name and purpose of the classes

2) The member variables and the functions of the class

3) Show the interactions between classes (for example, ownership or dependency)

4) Any relevant notes that don’t fit into the previous categories can be added

5) Numbering in relationships

b. For the “Perform Encryption Using an Encryption algorithm” and “Perform Decryption using a Decryption Algorithm” features, create System Sequence Diagrams. **Make sure** to include function names, data types, and ordering information. Feel free to add explanatory notes. To get extra help, check out the following links for some additional examples:

<http://en.wikipedia.org/wiki/Sequence_diagram>

<http://en.wikipedia.org/wiki/System_Sequence_Diagram>

c. You **must** implement TranspositionCipher and the SubstitutionCipher as classes. Hint: Inheritance could be very useful in order to create a generic parent class that both ciphers inherit from!

**3.** **Testing (2 artifacts)**: Develop a **driver** and show the driver:

1) For the system at large. In other words, describe inputs for “nominal” usage. You may need several scenarios. Also, suggest scenarios for abnormal usage and show what your program should do (for example, entering a negative number for a menu might ask the user to try again).

2) For each object. (Later, these tests can be automated easily using a simple driver function in the object)

*4. Implement your solution*

5. Test Results: *After developing your solution*, actually try all of your test cases (both system and unit testing). Actually show the results of your testing (a copy and paste from your program output is fine – don’t stress too much about formatting as long as the results are clear). You should have test results for every test case you described. If your system doesn’t behave as intended, you should note this. Note: Driver output will substitute for this phase.

|  |  |  |
| --- | --- | --- |
| Top of Form   |  |  | | --- | --- | | |  | | --- | | **2. How does the Honors component differ from normal course expectations (a copy of the syllabus may be provided to illustrate)? Address if the Honors component will be factored into the final grade in the course and, if so, what the grading expectations will be.** | |   Bottom of Form |

In the normal COMP2710 course, the students only work on two projects. In this honors course, the student will be working on the third project. A typical workload for the third project is 36 hours, assuming the student spend 9 hours a week and four weeks on this extra project.