Task:

This assignment requires implementation of a programmatic solution to hashing, hash tables and collision resolution. In part 2 you are asked to conduct research and prepare a report on encryption.

Part 1: Hashing

Create a console application to compare the time required to conduct searching activities on using a hash table.

You are tasked to build a console-based spell checker that will be used to show the spelling mistakes in a standard text document. You should use the same program framework from assignment #4.

Requirements:

* The application will start by reading all the words in a dictionary file that has been provided for you. The file is not a complete dictionary. It simply contains all the correct spelling for specific words in alphabetic order. The dictionary words will be the basis for a custom hash table in your application.
* By programmatic means you must store each of the dictionary words in your hash table. When the table is complete you must display it for review. You must consider the data being stored and devise your own hashing function and collision resolution mechanism. The suggest approach is to use some form of modular arithmetic combined with collision resolution by chaining, but ultimately it is your choice. Be prepared to fully explain your choice and algorithm.
* Reusing the code from part 2 of the binary search tree assignment, create the infrastructure to conduct the searching portion of the spell check, replacing the binary search tree with your hashing table and function.
* Once the hash table has been filled, you will read in a second file that is a sample document needing spell checking. You will then compare each word in the document against the hashed words.
  + If the search fails, you will display the misspelled word to the console.
* **Hint**: You cannot modify either the provided dictionary file or document file. To simplify the timing capture, consider reading each word of the target file to spell check in to a simple array and use the array to source the searches. Then use a second array to capture the misspelled words for later display. This should provide the needed isolation.

Part 2: Encryption

Conduct personal research in to two encryption techniques of your choice within these parameters:

* One technique must be a symmetrical encryption approach
* One technique must be an asymmetric encryption
* Describe each technique
* Compare the two algorithms
* Describe in your own words how the technique of hashing data may be used in encryption.

Evaluation:

This assignment is worth 43 marks. Please see the marking rubric below.

Assignment Notes:

The assignment must be demonstrated to the instructor on or before the due date during class.

If your assignment is late, please submit it to the Bright Space drop box for the assignment. This drop date will constitute the timestamp for evaluating any late penalty the assignment may incur.

See the **Marking Rubrics** below.

| Criteria – Part 1 | Marginal | Developing | | Good | | Exceptional | Marks |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | | 2 | | 3 |
| Hash Algorithm | * Hash table not used to store dictionary * too many errors exist | * Dictionary was correctly read into hash table - can be viewed on the console * Some errors exist | * Dictionary hash table was read and displayed correctly * Few errors exist | | | * Hash table contains all dictionary words and all spelling errors identified * Hash algorithm can be explained - can easily view table and any chains on the console * No bugs | **\_\_\_\_x3\_** |
| Collision Resolution |  | * An attempt was made to resolve collisions * Some errors exist | * Collision resolution reuses earlier classes built for this course. * Few errors exist | | | * Collision resolution reuses earlier classes built for this course and performs without bugs | **\_\_\_\_x2\_** |
|  |  |  | |  | | **Sub Total** | **\_\_\_\_ 15** |
| Spell Checker Output | * The program does not output misspelled words from the test document | * Some misspelled words are correctly identified and listed * Could not handle punctuation, capitalization or special characters * Some errors exist | * All misspelled words are correctly identified and listed * No errors | | |  | **\_\_\_\_** |
| Aesthetics | * incorrect or non-existent use of whitespace in output * output is confusing and hard to follow | * fair use of   whitespace   * most output is clear, but poorly presented | | * excellent use of whitespace * output is clear and attractively presented | |  | \_\_\_\_\_ |
| Readability | * source code is poorly organized and very difficult to read | * source code is fairly easy to read, but is hard to follow in some areas | | * source code is exceptionally well organized and easy to follow | |  | \_\_\_\_\_ |
| Reusability | * source code cannot be reused * no functions or classes used | * portions of code could be reused with modifications | | * source code could be easily reused with few modifications | |  | \_\_\_\_\_ |
| Efficiency | * contains large portions that could have been easily reduced using a different method * too much code is replicated, copy /pasted | * tried some methods to improve efficiency * can explain what they attempted | | * very clean and efficient code * can propose new ideas for improvement | |  | \_\_\_\_\_ |
| Comments | * little to no comments used | * comments are used, some are meaningful and easily understood * some files and functions have headers | | * not over/under commented * comments are meaningful and easily understood * files and functions have headers * Code is self-documenting | |  | \_\_\_\_\_ |
| **Naming** Convention | * no standard naming convention followed | * a standard naming convention was used for part of the program, but deviated often | | * industry standard naming convention used throughout the program | |  | \_\_\_\_\_ |
| Consistency | * no consistency in formatting or layout of source code | * source code formatting was present but inconsistent with whitespace, brackets, etc | | * source code formatting never deviated from the programmer’s layout | |  | \_\_\_\_\_ |
| SubTotal | | **16** |
| Task 1 Total | | **31** |

**Marking Rubric – Task 2**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Exceptional** | **Acceptable** | **Below Standard** |
| **Organization** | Extremely well organized; logical format that was easy to follow; flowed smoothly from one idea to another and cleverly conveyed; to enhance the presentation (2) | Somewhat organized; ideas were not presented coherently and transitions were not always smooth, which at times distracted the reader (1) | Choppy and confusing; format was difficult to follow; transitions of ideas were abrupt and seriously distracted the reader (0) |
| **Content Accuracy – Symmetric Encryption** | Completely accurate; all facts were precise and explicit (2) | Somewhat accurate; more than a few inconsistencies or errors in information (1) | Completely inaccurate; the facts in this project were misleading to the reader (0) |
| **Content Accuracy – Asymmetric Encryption** | Completely accurate; all facts were precise and explicit (2) | Somewhat accurate; more than a few inconsistencies or errors in information (1) | Completely inaccurate; the facts in this project were misleading to the reader (0) |
| **Content Accuracy – Use of Hashing in Encryption** | Completely accurate; all facts were precise and explicit (2) | Somewhat accurate; more than a few inconsistencies or errors in information (1) | Completely inaccurate; the facts in this project were misleading to the reader (0) |
| **Research** | Went above and beyond to research information; utilized many different resources to make project effective (2) | Used the material provided in an acceptable manner, but did not consult many additional resources (1) | Did not utilize resources effectively; did little or no fact gathering on the topic (0) |
| **Polish** | Flawless grammar and punctuation (2) | Some grammar and punctuation errors. (1) | More than 20 grammar and punctuation errors. (0) |
|  |  | **Task Total** |  |
|  |  | **(of 12)** |  |
|  |  |  |  |
|  |  | **Assignment Total** |  |
|  |  | **(of 43)** |  |
|  |  |  |  |

0 - Assignment not submitted or work not original.