**BACHELOR OF COMPUTER SCIENCE**

**FACULTY/SCHOOL OF SCHOOL OF COMPUTER SCIENCE**

**BINA NUSANTARA UNIVERSITY**

**JAKARTA**

**ASSESSMENT FORM**

**Course: COMP6048001 - Data Structure**

**Method of Assessment:** **Case Study**

**Semester/Academic Year : 2/2024-2025**

**Name of Lecturer : ………………………**

**Date : ………………………**

**Class : ………………………**

**Topic : Tries**

|  |  |
| --- | --- |
| **Group Members :** | 1\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  2\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  3\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  4\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  5\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  6\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  7\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  8\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

**Student Outcomes:**

**SO 2 - Mampu merancang, mengimplementasikan, dan mengevaluasi solusi berbasis komputasi untuk memenuhi serangkaian persyaratan komputasi dalam konteks ilmu computer**

***Able to design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of computer science***

**Learning Objectives:**

**LObj 2.2 - Mampu mengimplementasikan solusi berbasis komputasi untuk memenuhi serangkaian persyaratan komputasi tertentu dalam konteks ilmu computer**

***Able to implement a computing-based solution to meet a given set of computing requirements in the context of computer science***

**Learning Outcomes:**

**LO 3 - Apply data structures using C**

| **No** | **Related LO-LOBJ-SO** | **Assessment criteria** | **Weight** | **Excellent (85 - 100)** | **Good (75-84)** | **Average (65-74)** | **Poor (0 - 64)** | **Score** | **(Score x Weight)** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | LO 3 – LObj 2.2 – SO 2 | Ability to design an algorithm for the problem | **80 %** | Able to solve 76 - 100% the problem with fully functional feature. | Able to solve 51-75% the problem and lack some features. | Able to solve 26-50% of the problem and lack some features. | Able to solve less than 25% of the problem. |  |  |
| 2 | LO 3 – LObj 2.2 –SO 2 | Ability to explain the algorithm and make a documentation | **20 %** | Able to explain the algorithm well. The documentation is good and complete. | Able to give a good explaination of the algorithm. The documentation is good enough. | Able to give an enough explaination of the algorithm The documentation is good enough, but it is not complete. | The explanation the algorithm is bad. The documentation is not complete. |  |  |
|  |  | **Total Score:** ∑(Score x Weight) | | | | | | |  |

Remarks:

**ASSESSMENT METHOD**

**Instructions:**

* This case study is individual project.
* The case study scoring consist of two parts:
  1. **[LO3-LObj 2.2-SO 2, 80 points]** Design an algorithm in **cpp** to solve the problem.
  2. **[LO3-LObj 2.2-SO 2, 20 points]** Create a documentation for your program in PDF (Explain the logic, purpose, and maybe limitation behind each line of the code/function with human language), **please write the code, don’t just screenshot it**. Test your program with **custom cases that you made** and put the custom cases along with the result (**screenshots**) in the documentation PDF.
     + Example code explanation *may* look like this (**Explaining each line of code**):

|  |
| --- |
| include<stdio.h> // included for standard input output function  int main(){  // declare variable number as integer  int number;  // Ask for input number from 1 to 10. This code will loop while the number inputted is either less than 1 or more than 10.  do{  printf(“Enter number between 1 & 10: ”);  scanf(“%d”, &number);  }while(number < 1 || number > 10);  return 0;  } |

* + - Example code explanation for function *may* *also* look like this (**Explaining how a function works**):

|  |
| --- |
| int biggerNumber(int a, int b){  int bigger = (a > b) ? a : b;  return big  } |

The code above is used to find a bigger number from 2 numbers which are in the parameter: ‘a’ and ‘b’. To find the bigger number, I simply use a ternary operator which has the condition if ‘a’ is more than ‘b’, then ‘a’ is the bigger number, else then ‘b’ is the bigger number, I assign either value in a temporary variable ‘bigger’, then return it. By using this logic, the function will return ‘b’ if the numbers have the same value, which is okay in our program.

* + - The custom case **(that you made yourself)** should **at least** have:
      * **Input of 15 slang words**,
      * **Search 5 words**,
      * **View prefix 5 words** and
      * **View all**.
* Submit the .**cpp** and **PDF** files to Binusmaya (zip them).

**Note for Lecturers**:

* The lecture notifies this case study to the student from Week 1.
* Deadline for the case study *ideally* is in week 9. However, the deadline set in Binusmaya will be in the last week.
* The student should submit the report to binusmaya no later than deadline.
* **If the students do plagiarism, their score for this case study will be 0 (zero). The lecturers have the privilage to determine whether the students do plagiarism.**

## Soal

*Case*

**Boogle**

Boogle is a company that create, and document new slang words based on the internet. You as a programmer working at the Boogle company are asked to create an application that is useful for seeing what slang words have been released by Boogle. **Ensure that you didn’t use regex in your work, or it will affect your score. Please focus on the main logic and main feature! (Design are not scored)**. The requirements are:

* The application consists of **5 menus**:

1. **Release a new slang word**
2. **Search a slang word**
3. **View all slang words starting with a certain prefix word**
4. **View all slang words**
5. **Exit**

* If user choose menu **1** (“**Release a new slang word**”), then the program will:
* Ask the user to **input** the **new slang word**. Validate that the **slang word** must be **more than 1 character and contains no space**.
* Ask the user to **input** the **description (meaning)** of the new word. Validate that the **description** must be **more than 1 word**.
* **Store** the new released slang word to a **Trie** data structure along with its description.

|  |
| --- |
| Input a new slang word [Must be more than 1 characters and contains no space]: d  Input a new slang word [Must be more than 1 characters and contains no space]: d a  Input a new slang word [Must be more than 1 characters and contains no space]: da  Input a new slang word description [Must be more than 2 words]: The  Input a new slang word description [Must be more than 2 words]: The word the  Successfully released new slang word.  Press enter to continue... |

* If the slang word **already exists** in the Trie, then **update** the description with the new description.

|  |
| --- |
| Input a new slang word [Must be more than 1 characters and contains no space]: da  Input a new slang word description [Must be more than 2 words]: Same meaning as word “the”  Successfully updated a slang word.  Press enter to continue... |

* If user choose menu **2** (“**Search a slang word**”), then the program will:
* Ask the user to **input** the **slang word that want to be searched**. Validate that the **slang word** must be **more than 1 character and contains no space**.
* **Search** the input word in the Trie data structure.
* If there **is no such word**, please **show empty message** for the user and go back to main menu.

|  |
| --- |
| Input a slang word to be searched [Must be more than 1 characters and contains no space]: de  There is no word “de” in the dictionary.  Press enter to continue... |

* If there **is such word**, please **show the word along with its description**.

|  |
| --- |
| Input a slang word to be searched [Must be more than 1 characters and contains no space]: d  Input a slang word to be searched [Must be more than 1 characters and contains no space]: d a  Input a slang word to be searched [Must be more than 1 characters and contains no space]: da  Slang word : da  Description : Same meaning as word “the”  Press enter to continue... |

* If user choose menu **3** (“**View all slang words starting with a certain prefix word**”), then the program will:
* Ask the user to **input** the **prefix word that want to be searched**.
* **Search** the input word in the Trie data structure.
* If there **is no such word**, please **show empty message** for the user.

|  |
| --- |
| Input a prefix to be searched: de  There is no prefix “de” in the dictionary.  Press enter to continue... |

* If there **is such word**, please **show the list of words in the dictionary that starts with the prefix word in lexicographical order.**

|  |
| --- |
| Input a prefix to be searched: da  Words starts with “da”:  1. da  2. dadan  3. daijobu  4. dawg  5. dazz  Press enter to continue... |

* If user choose menu **4** (“**View all slang words**”), then the program will:
* If there **is no word yet** in the dictionary, please **show empty message** for the user.

|  |
| --- |
| There is no slang word yet in the dictionary.  Press enter to continue... |

* Else, please **show the list of all words in the dictionary** in **lexicographical order.**

|  |
| --- |
| List of all slang words in the dictionary:  1. da  2. dadan  3. daijobu  4. dawg  5. dazz  6. rizz  7. simp  Press enter to continue... |

* If user choose menu **5** (“**Exit**”), then the program will be closed.

|  |
| --- |
| Thank you... Have a nice day :) |