|  |  |
| --- | --- |
| **Onsite Case** | Description: D:\logo binus baru.png |
| DSH4 |
| **Periode Berlaku** Semester Ganjil 2015/2016  ***Valid on*** *Odd Year 2015/2016* | **Software Laboratory Center**  **Assistant Recruitment 16-1** |

## Materi

*Material*

* Binary Tree Implementation
* Insert Binary Tree
* Pop 1 node
* Pop All
* Searching in Binary Tree

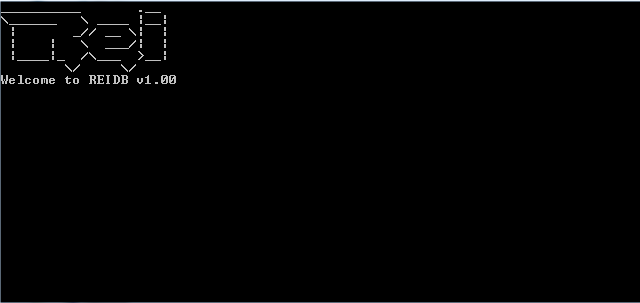
## Soal

*Case*

**ReiDB**

**ReiDB** is a program to store the data of Authors and their books. The program store the data in the structure of **Binary Search Tree**(**BST**). **ReiDB** is developed with C Programming Language. As a programmer, you are asked to make a similar program with the following criterias :

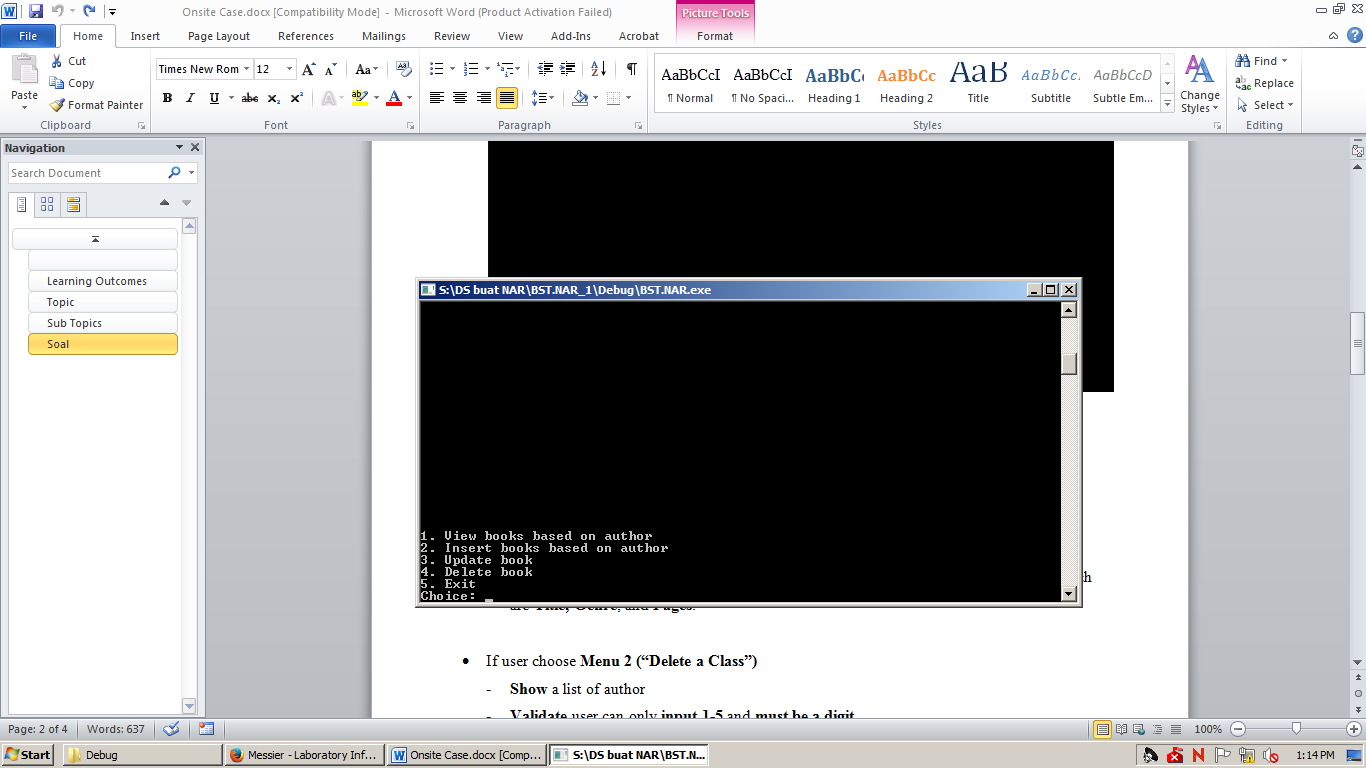
* First, The program will show splash screen.



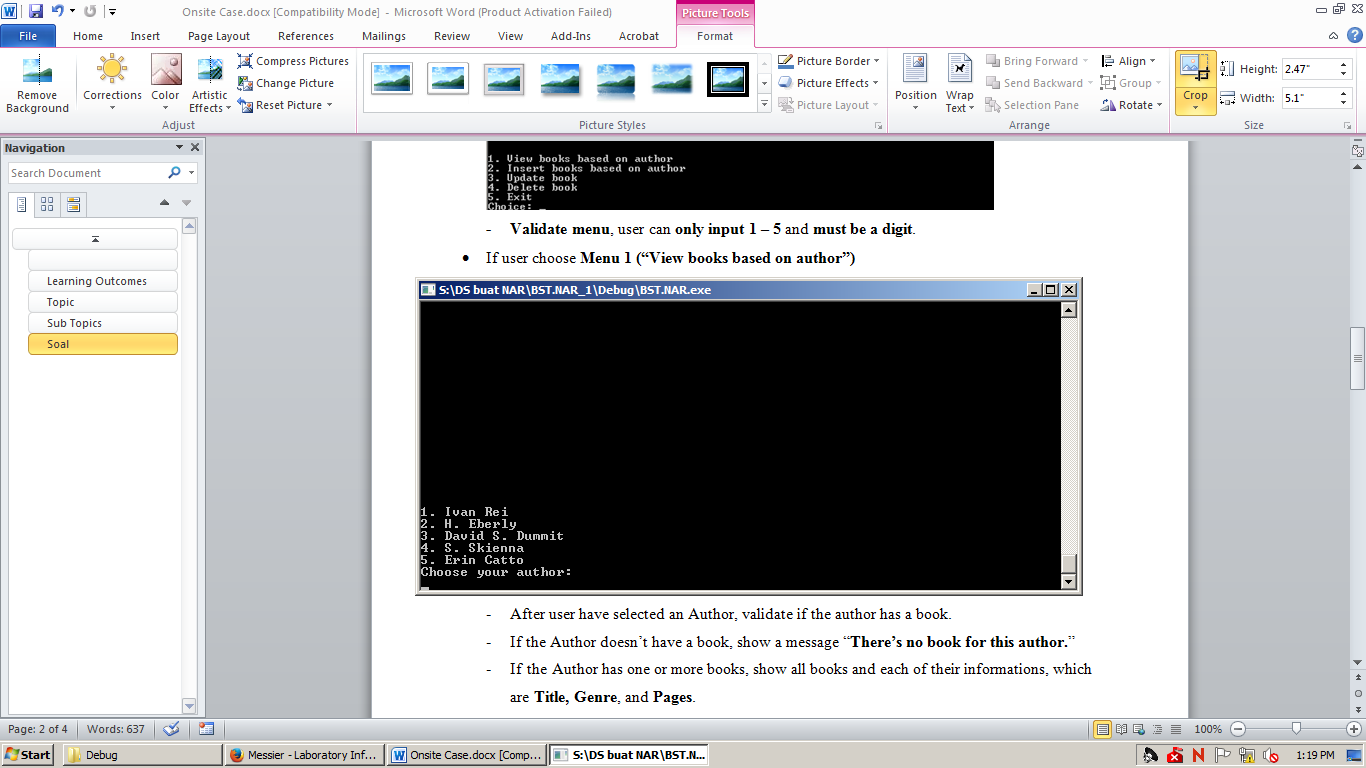
* There’re 5 **main menus** :

1. **View books based on author**
2. **Insert books based on author**
3. **Update book**
4. **Delete book**
5. **Exit**

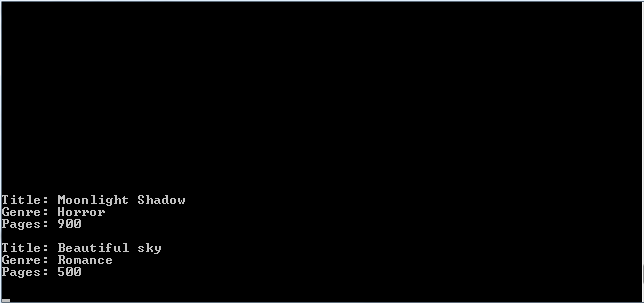
* Validate **Menu** must be **numeric** and between 1 to 5.



**1. View books based on author (Menu 1)**

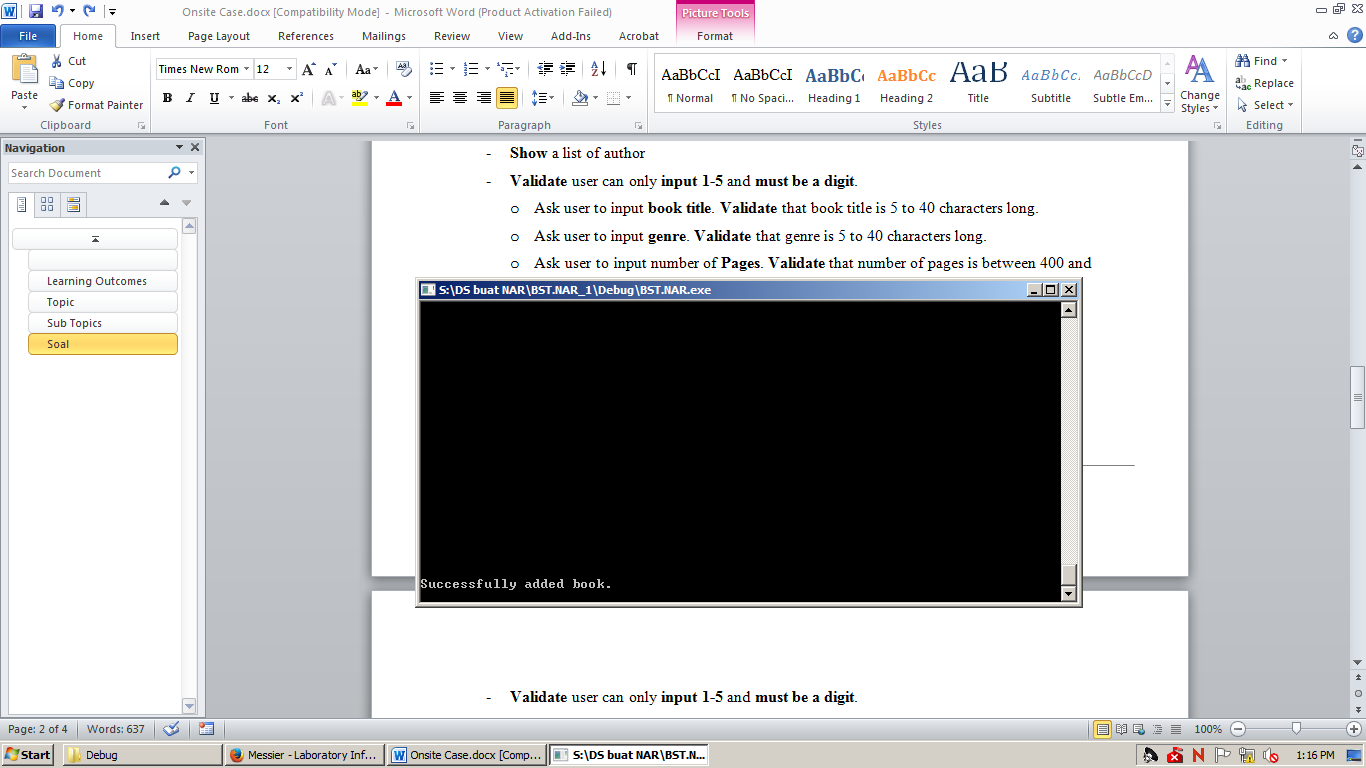


* **Display** a list of author.
* Then, the program will ask user to choose author, **validate** user can only **input between 1-5** and **must be a digit**.
* After user have selected an Author, validate if the author has a book.
* If the Author doesn’t have a book, show a message “**There’s no book for this author.**”
* If the Author has one or more books, show all books and each of their informations **In Orderly**, which are **Title, Genre**, and **Pages**.

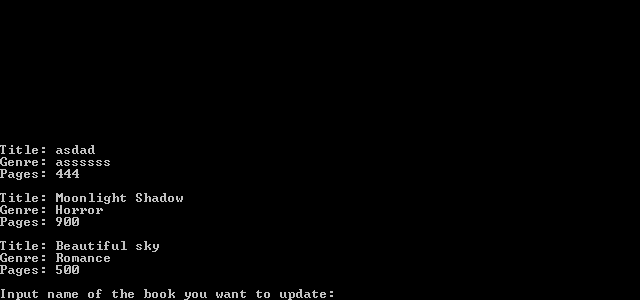
****

**2. Insert book based on author (Menu 2)**

* **Display** a list of author
* Then, the program will ask user to choose author, **validate** user can only **input between 1-5** and **must be a digit**.
* Program will ask user to input **book title**. **Validate** book title must **between** **5 to 40** **characters** long.
* Program will ask user to input **genre**. **Validate** genre must **between 5** **to 40 characters** long.
* Program will ask user to input number of **Pages**. **Validate** number of pages **must between 400 and 1000**.
* After that, display a message “**Successfully added book**”.



**3. Update Book (Menu 3)**

****

* **Display** a list of author.
* Then, the program will ask user to choose author, **validate** user can only **input between 1-5** and **must be a digit**.
* **Show** list of books and its information that belong to the Author, then ask the user to input **name** of the book to be updated.
* **Validate** that user input **must matches** one of Author’s book’s name. Otherwise, show a message “**No book with title you specified**”. Then go back to main menu.
* If success, ask user to input **new title**, **new genre**, and **new number of pages**.
* **Validate** new book title must **between** **5 to 40** **characters** long.
* **Validate** new genre must **between 5** **to 40 characters** long.
* **Validate** new number of pages **must between 400 and 1000**.
* After that, **the book will be updated**.

**4. Delete Book (Menu 4)**

****

* **Display** a list of author.
* Then, the program will ask user to choose author, **validate** user can only **input between 1-5** and **must be a digit**.
* **Show** list of books and its information that belong to the Author. Then, ask user to input book title to be deleted.
* **Validate** that user input **must matches** one of the Author’s book. Otherwise, show a message “**Book not found**”. Then go back to main menu.
* If success, the book which title matches with user input will be deleted. Show a message “**Successfully deleted book**”.

**5. Exit (Menu 5)**

* The program will automatically close itself.

**\*) Note :**

* for every aspect you have to use **dynamic memory allocation** to get maximum score.
* Remember only to use **BST** and its rules.
* **Primary key** used for **comparison** is **book title** for each Author’s.

**Please run the EXE file to see the sample program.**