# Birthday and Anniversary Database

# Basic of Programming 2

# Final Project

# Documentation

# Programmer's Guide


Name: Mean Diamand

Neptun Code: KSG25Z

## Introduction

A database is a collection of organized data that can be accessed, managed, and updated in many ways. It is used to store and retrieve information and is accessed by multiple users or applications simultaneously. This program will show how the database work based on Object Oriented Programming with inheritance and multimap. This documentation will provide a description and explanation of this database program which will guide the users and developer about how this program works.

Birthday and Anniversary database is a database that stores the information of the event of the users, for example, their birthday, anniversary, nameday, or marriage. The users will be able to add, delete, search, and list all events from this database willingly.

## Design

As I have mentioned above, Birthday and Anniversary database was designed based on the Object-Oriented Style of Programming that is taught in the Basic of Programming 2 course in BME. Many techniques were used in this program such as:

- Dynamic memory management(new/delete): used for allocating the memory for storing the events that are added and releasing it after finished use.
- Exception handling(try/catch): used for checking the runtime error that may occur when opening the text file that we used to store the data after the program is terminated.
- File management: used for reading the data from and writing the data into the text file for saving after terminating the program.
- Inheritance: used for creating a based class and the derived classes that inherit all the properties of the based class. There will be a based class called Event where its derived classes are Birthday, Anniversary, Nameday, and Marriage.
- Polymorphism: used for creating virtual functions that are declared in the based class and can be overridden in its derived class.

- Operator overloading: used for creating a custom operator that works with the class.

This database program consists of many files such as:

- main.cpp: contains the switch case that asks the user to input the choice for choosing the options of going into the database menu, checking the instructions, checking the credit of the developer, or exiting the program.
- option.h: contains all the function declarations of the functions that are related to adding, deleting, searching, and printing the data from the database. Also functions for reading the text file, introduction, loading, and saving the data into the database.
- option.cpp: contains all the implementation of those functions that were declared in the option.h header file.
- event.h: contains the Event class which is a based class, operator overloading, and the attributes and methods of it.
- event.cpp: contains all the implementation of those methods and operator overloading that were declared in the class Event of the event.h header file.
- birthday.h: contains the Birthday class which is a derived class of the Event class, operator overloading, and the attributes and methods of it.
- birthday.cpp: contains all the implementation of those methods and operator overloading that were declared in the class Birthday of the birthday.h header file.
- anniversary.h: contains the Anniversary class which is a derived class of the Event class, operator overloading, and the attributes and methods of it.
- anniversary.cpp: contains all the implementation of those methods and operator overloading that were declared in the class Anniversary of the anniversary.h header file.
- nameday.h: contains the Nameday class which is a derived class of the Event class, operator overloading, and the attributes and methods of it.
- nameday.cpp: contains all the implementation of those methods and operator overloading that were declared in the class Nameday of the nameday.h header file.
- marriage.h: contains the Marriage class which is a derived class of the Event class, operator overloading, and the attributes and methods of it.
- marriage.cpp: contains all the implementation of those methods and operator overloading that were declared in the class Marriage of the marriage.h header file.

## Implementation:

1 Based Class:

- Class Event consisting of:
  + 2 private attributes:
    - category – string data type for storing the category of the users such as friend, family, relative, or colleague.
    - name – string data type for storing the name of the users.
  + 1 default constructor: for initializing the object of the class.
  + 2 setter methods: void type methods that receive a parameter for setting the content into the attribute.
    - set_category()
    - set_name()
  + 2 getter methods: string type methods that return a parameter which is the content stored in the attribute. It is also set to be a constant method so that the content of the object won't be modified.
    - get_category()
    - get_name()
  + 13 virtual functions: that can be overridden by the derived classes.
  + 2 operator overloading: for customize the operator to work with the object of the class.
    - <<: for printing the information of the object of the class.
    - >>: for scanning the information of the object of the class.

4 Derived Classes:

- **Class Birthday** is a derived class of the Event Class which consists of:
  + 3 private attributes:
    - year – integer data type for storing the year of the event.
    - month – integer data type for storing the month of the event.
    - day – integer data type for storing the day of the event.

+ 1 default constructor: for initializing the object of the class by calling the constructor of the based class to get the parameter of the attributes from the based class and initialize the attributes of the derived class itself.

+ 3 setter methods: void type methods that receive a parameter for setting the content into the attribute.

  - set_year()
  - set_month()
  - set_day()

+ 3 getter methods: integer type methods that return a parameter which is the content stored in the attribute. It is also set to be a constant method so that the content of the object won't be modified.

  - get_year()
  - get_month()
  - get_day()

+ print_event(): output steam reference type method that prints the date of the event including the month, day, and year.

+ input_event(): void type method that get the input of the date and set it into the attributes.

+ save_database(): void type method that saves the information of the attributes of this class and its based class.

+ load_database(): void type method that loads the information and set that information into the attributes.

+ == operator overloading: bool type method that comparing between two events of the class.

- **Class Anniversary** is a derived class of the Event Class which consists of:
  + 4 private attributes:

- Partner – string data type for storing the name of the partner of the event's owner.
- year – integer data type for storing the year of the event.
- month – integer data type for storing the month of the event.
- day – integer data type for storing the day of the event.

+ 1 default constructor: for initializing the object of the class by calling the constructor of the based class to get the parameter of the attributes from the based class and initialize the attributes of the derived class itself.

+ 4 setter methods: void type methods that receive a parameter for setting the content into the attribute.

- set_partner()
- set_year()
- set_month()
- set_day()

+ 4 getter methods: string and integer type methods that return a parameter which is the content stored in the attribute. It is also set to be a constant method so that the content of the object won't be modified.

- get_partner()
- get_year()
- get_month()
- get_day()

+ print_event(): output steam reference type method that prints the name of the partner of the event's owner and the date of the event including the month, day, and year.

+ input_event(): void type method that get the input of the date and set it into the attributes.

+ save_database(): void type method that saves the information of the attributes of this class and its based class.

+ load_database(): void type method that loads the information and set that information into the attributes.

+ == operator overloading: bool type method that comparing between two events of the class.

- **Class Nameday** is a derived class of the Event Class which consists of:

+ 3 private attributes:
    - year – integer data type for storing the year of the event.
    - month – integer data type for storing the month of the event.
    - day – integer data type for storing the day of the event.

+ 1 default constructor: for initializing the object of the class by calling the constructor of the based class to get the parameter of the attributes from the based class and initialize the attributes of the derived class itself.

+ 3 setter methods: void type methods that receive a parameter for setting the content into the attribute.

    - set_year()
    - set_month()
    - set_day()

+ 3 getter methods: integer type methods that return a parameter which is the content stored in the attribute. It is also set to be a constant method so that the content of the object won't be modified.

    - get_year()
    - get_month()
    - get_day()

+ print_event(): output steam reference type method that prints the date of the event including the month, day, and year.

+ input_event(): void type method that get the input of the date and set it into the attributes.

+ save_database(): void type method that saves the information of the attributes of this class and its based class.

+ load_database(): void type method that loads the information and set that information into the attributes.

+ == operator overloading: bool type method that comparing between two events of the class.

- **Class Marriage** is a derived class of the Event Class which consists of:

+ 4 private attributes:
    - Partner – string data type for storing the name of the partner of the event's owner.
    - year – integer data type for storing the year of the event.
    - month – integer data type for storing the month of the event.
    - day – integer data type for storing the day of the event.

+ 1 default constructor: for initializing the object of the class by calling the constructor of the based class to get the parameter of the attributes from the based class and initialize the attributes of the derived class itself.

+ 4 setter methods: void type methods that receive a parameter for setting the content into the attribute.

    - set_partner()
    - set_year()
    - set_month()
    - set_day()

+ 4 getter methods: string and integer type methods that return a parameter which is the content stored in the attribute. It is also set to be a constant method so that the content of the object won't be modified.

    - get_partner()
    - get_year()
    - get_month()

- get_day()

+ print_event(): output steam reference type method that prints the name of the partner of the event's owner and the date of the event including the month, day, and year.

+ input_event(): void type method that get the input of the date and set it into the attributes.

+ save_database(): void type method that saves the information of the attributes of this class and its based class.

+ load_database(): void type method that loads the information and set that information into the attributes.

+ == operator overloading: bool type method that comparing between two events of the class.

- 18 Global Functions

+ introduction(): integer type function that returns the input choice of the users when they wanted to select the option in the menu, and it is also used for reading a text file that contains the introduction of my program. In this function, I implemented a file-handling method for reading a text file.

+ read_the_file(): a void function that is used for reading the text file that is wanted to be displayed on the console by using the file handling method for reading the text file.

+ database(): void function that is used to choose the options which are displayed on the databases menu that will be shown when the users start the databases. In this function, I implemented the switch case in the conditional statement method for the database options. It also contains the declaration of the multimaps that I used for storing the events and it is also where I load and save the information of the text file.

+ print_list_event(): a void function that is used for printing the list of events by iterating through the multimap.

+ delete_event(): void function that displays the list of the event for the user to check which event they want to delete then ask the user to input the name of the owner of the event that they want to delete then it will iterate through the multimap and find that event then delete it.

+ search_event(): a void function that asks the user to input the name of the owner of the event that they wanted to search for and it will iterate through the multimap and find that event then display it.

+ 4 loading functions: a void function that is used for opening the text file and reading the list of the event and inserting it into the multimap using the name as the primary key to notify each event in the map and then close the file.

- load_birthday()
- load_anniversary()
- load_nameday()
- load_marriage()

+ 4 adding functions: a void function that is used for creating new events and asks the user to input the information of the new event and then check this new event if it already existed in the map, if not then it will be inserted into the map.

- add_birthday()
- add_anniversary()
- add_nameday()
- add_marriage()

+ 4 saving functions: a void function that is used for opening the text file and saving every information from the map into the text file and releasing all the memory that was allocated when adding the event and then closing the file.

- save_birthday()

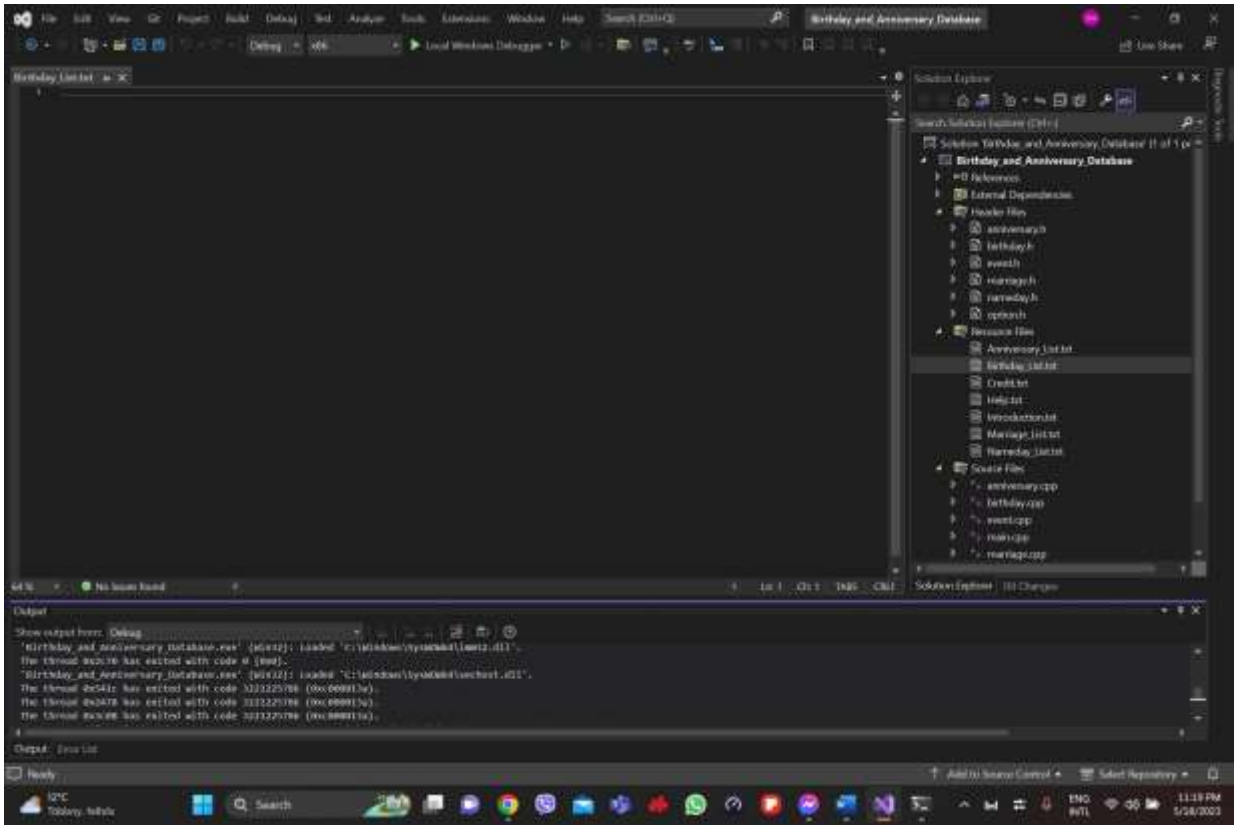
- save_anniversary()
- save_nameday()
- save_marriage()

- Exception Handling:
  + errorfile: used for throwing when the text file that stored the list of events cannot be opened. It was used in the database() function when we start the database and try to open the text file to load all the information into the map.

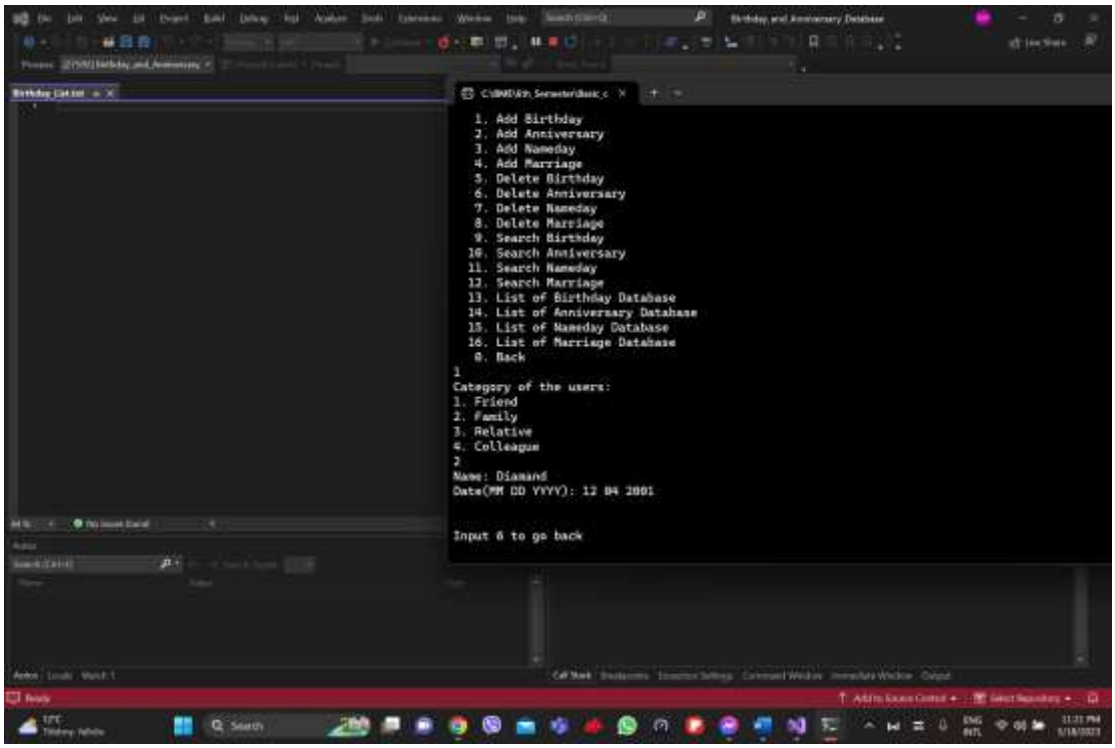
## Program Testing

+ Add and Save Test Case:

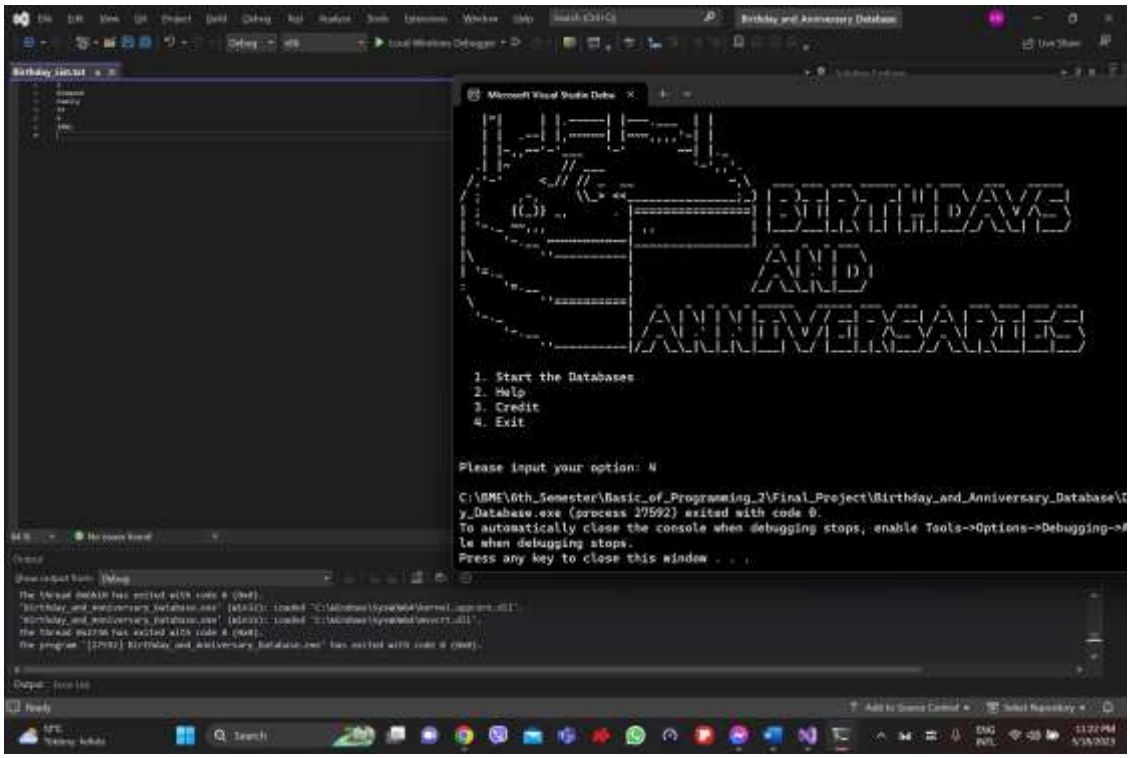
Initially, the text file that is used for storing the information of the events is empty as shown below.



Now, I'm inputting a new event into the database.

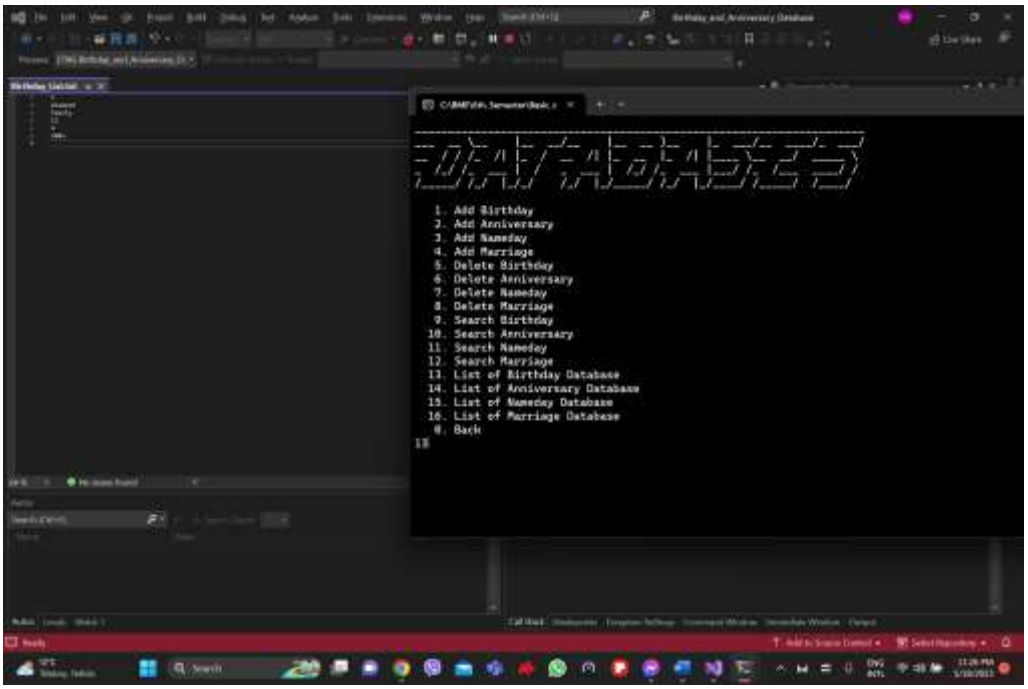



After terminating the program, the information of the event that I inputted into the database was added and saved into the Birthday_List.txt text file successfully.

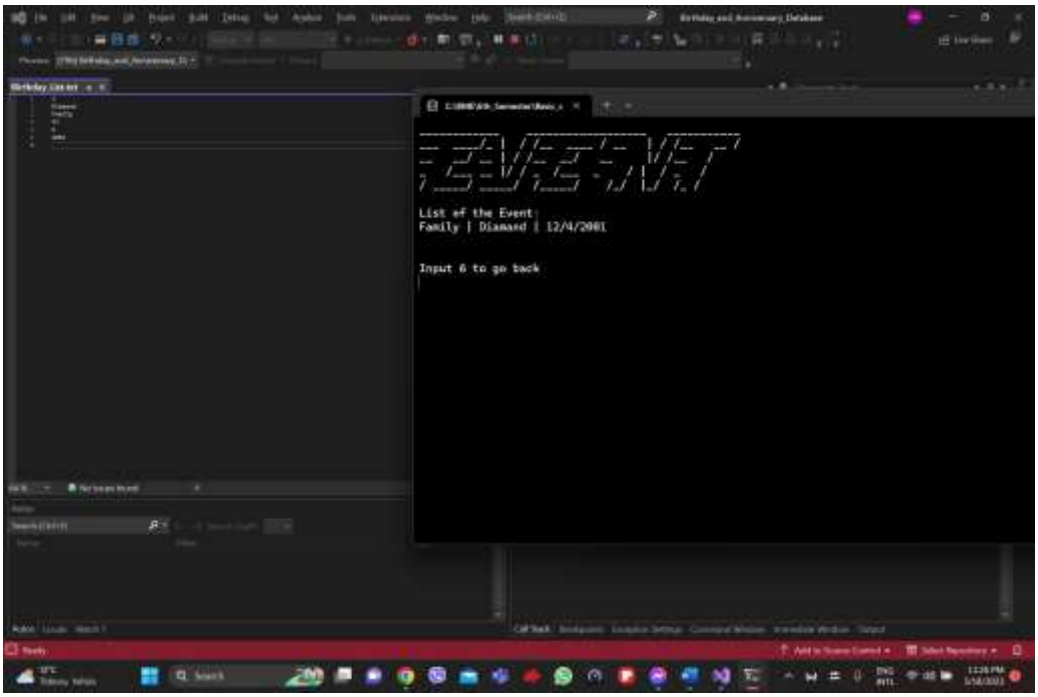


+ Load Test Case:

Now that, we already have one event stored inside the text file let's try to load it back into the program by listing all the Birthday events.

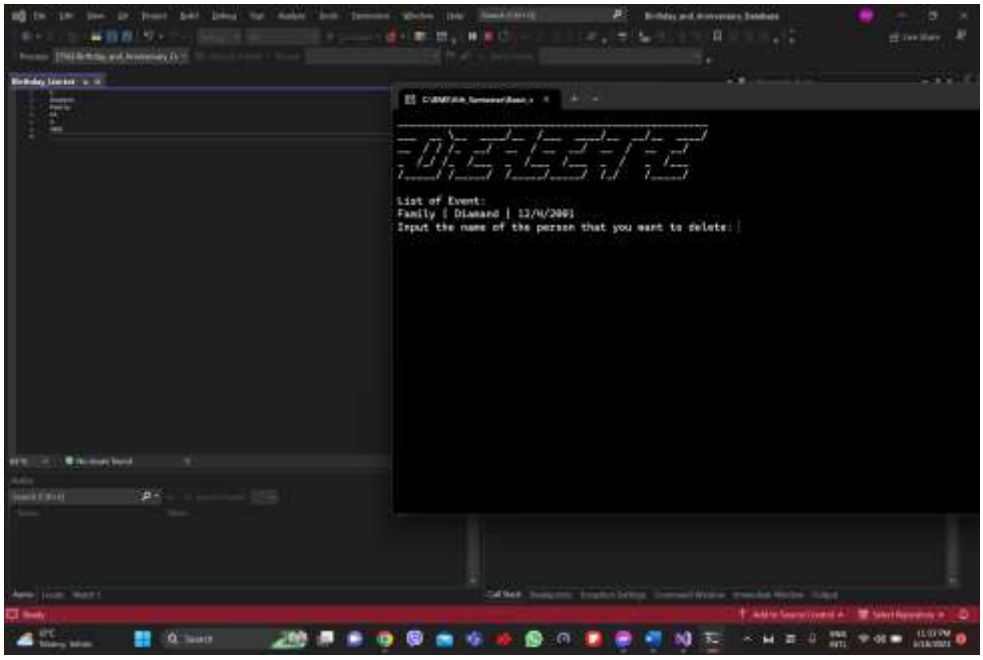



Now it seems like, the program managed to load the information back perfectly. As you can see the event stored inside the text file is appearing on the program.

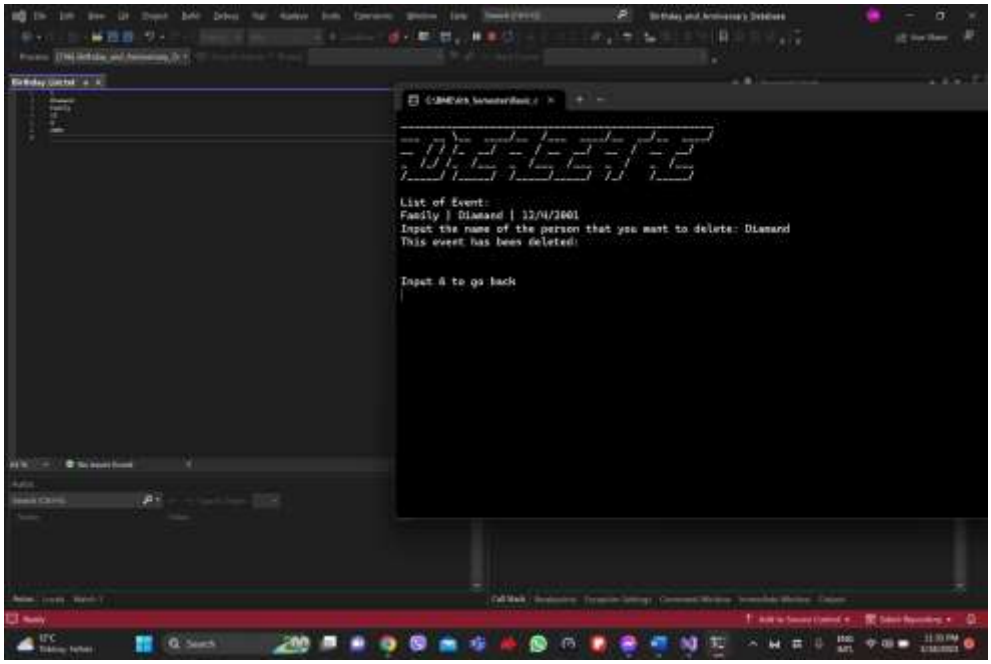


+ Delete Test Case:

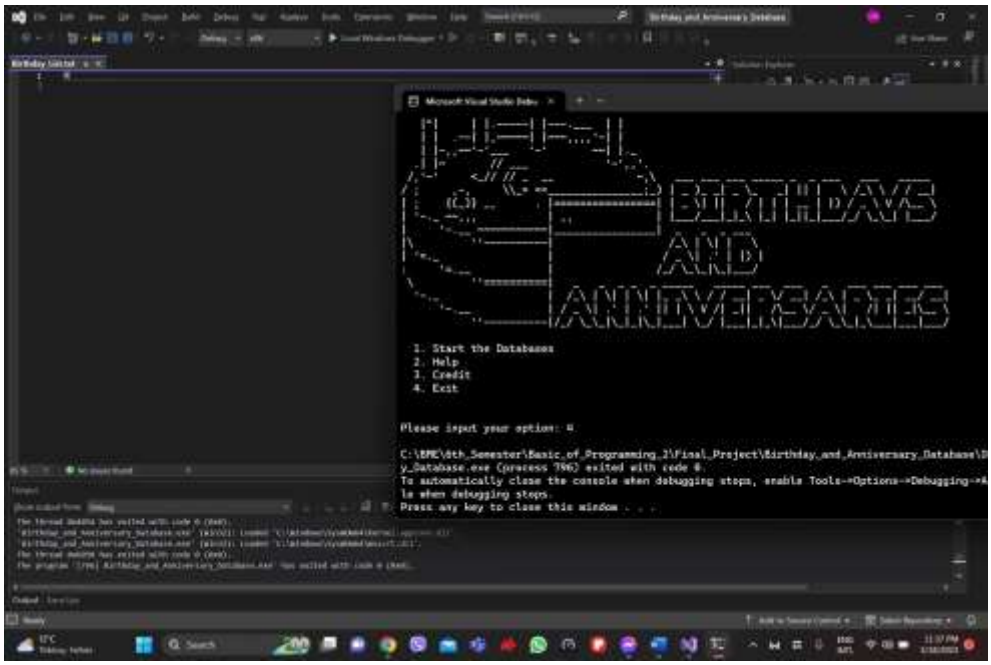
Now let's try to delete the event from the database. Here, you can see that the previous information that was added is shown below in the text file and is also displayed on the program.




Now I'm entering the name of the owner of the event, and the event will be deleted from the map then it will show an empty list.

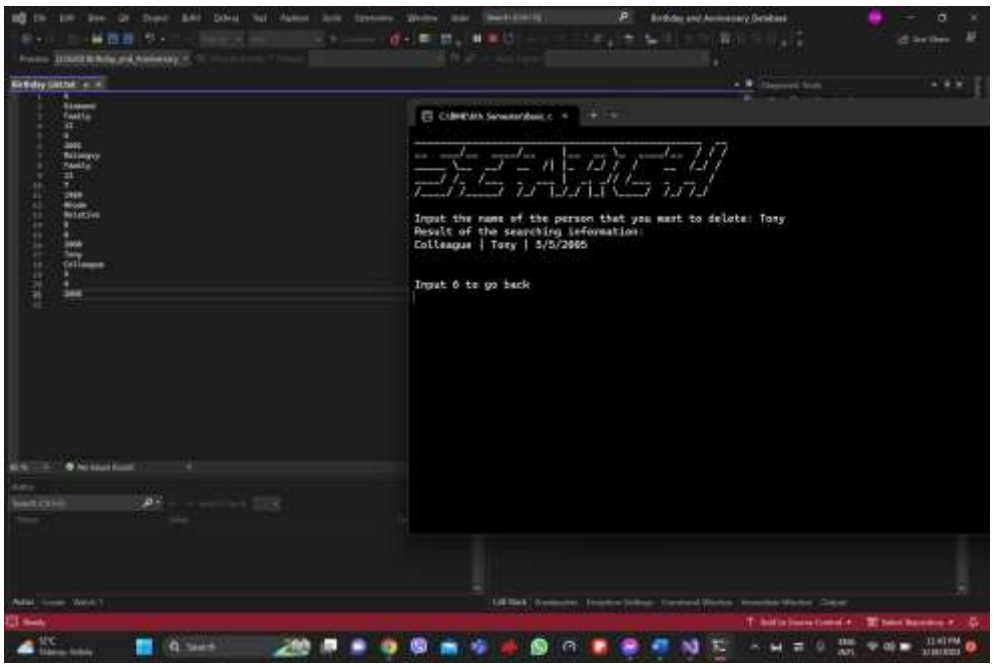


And after I terminated the program. The text file will update to 0 which means that the text file doesn't contain any information anymore which shows that the deletion is successful.




+ Query Test Case:

Now let's try to query the information from the database by searching the event using the primary key which is the name of the owner of the event and it seems like the query came out perfectly as we expected.



## Conclusion

After going through these 4 test cases from above, it seems like this database is working perfectly fine as expected with the fundamental features of the database which are adding, deleting, searching, and listing all the information of the database.