NAME: LIKHITH R

USN: 1NH18IS053

TITLE: ASTRO SCIENCE

CHAPTER-1

INTRODUCTION

This mini project “ASTRO SCIENCE” is based on the astrology prediction software in the digital life. The main aim of the project is to store details of different peoples in the same place. Which allows users to know their zodiac sign information instantly? The project demonstrates the creation of user interface of a system without the use of graphics applications. The project uses the basic functions of a python programming like pickle module, looping statements, string manipulation functions and dictionary to save and display the content. Here we can create a new database, append new elements and search the content from a dictionary.

1.1 PROBLEM DEFINATION :

1. An important purpose of astro science system is to streamline administrative task and reduce overhead by providing a single unified information about their zodiac sign and characters.
2. This software consist of different functions and modules which helps us to perform different operations in one place if this functions are not used in same software it will take more time for user to search or update the content of the dictionary.

1.2 PROPOSED SYSTEM :

1. Astro science can be used to manage or provide a zodiac sign related information.
2. This is the astro-centric databases that provides a fully integrated approach to tracking of all the information of the user.
3. This application provides different options for user to append ,search and display the different details from/in existing contact database.
4. This application mainly built to demonstrate the usage of the different python functions, modules and working of the dictionary data type with the lists and files.
5. A astro science is a software application or set of related programs that are used to create and manage digital content. This is mainly used for enterprise content management and web western astrology prediction.

1.3 OBJECTIVES :

1. To create a project using python programming and it’s features. Which allows users to implement a features like inbuilt modules ,control statements and user defiend module.
2. To learn about the different header files and conditions which is used to write a program.
3. Storing the data systematically to access it easily. Using the dictionary and by using pickling we store the data and by unpickling we access the data in the file.
4. It will save time for user by accessing the information about the people in one place.
5. This is used to document management and conversion management of the user.
6. Scheduling the appointments based on the database searching and storing.

1.3 METHODOLOGY:

In astro science I will be using Pickle module, dictionary, files, strings to implement the menu driven program. In will first display the menu content to user and asks the user to enter the choice .

Here above menu part consisting of creating a new file, appending of new elements and displaying of the content present in that file.

Creating new file erases the all data present in the file and try to add a new content to it, appending function is used to add a new element to the file if needed by the user, display function used to display the content and the search the content entered by the user if found display the result else error is display if user choice is 0 then it comes out of the program.

1.4 EXPECTED OUTCOMES :

```
WELCOME TO WESTREN ASTROLOGY
************MENU*************
1. CREATE NEW FILE
2. APPEND A NEW TOKEN
3. DISPLAY
4.SEARCH
0. EXIT
ENTER THE CHOICE:1
//CREATE NEW FILE
ENTER THE NUMBER OF TOKENS:1
ENTER YOUR NAME
DD/MM/YY
ENTER THE DATE OF BIRTH BY SPACING:
************MENU*************
1. CREATE NEW FILE
2. APPEND A NEW TOKEN
3. DISPLAY
4.SEARCH
0. EXIT
ENTER THE CHOICE:2
// APPENDING A NEW ELEMENT
ENTER THE NUMBER OF NEW TOKENS:1
ENTER YOUR NAME
DD/MM/YY
ENTER THE DATE OF BIRTH BY SPACING:
************MENU*************
1. CREATE NEW FILE
2. APPEND A NEW TOKEN
3. DISPLAY
4.SEARCH
```

```
0. EXIT
ENTER THE CHOICE:3
//DISPLAYING THE CONTENT
{‘ ‘ : [ ] , ‘ ‘ : [ ] }
dict_keys([‘ ‘ ],[’  ‘])
dict_values([[  ]],[[ ]])
ENTER THE NAME:
ENTER THE DATE OF BIRTH BY SPACING

************MENU*************

1. CREATE NEW FILE
2. APPEND A NEW TOKEN
3. DISPLAY
4.SEARCH
0. EXIT
ENTER THE CHOICE: 0
EXITING!
```

1.5:HARDWARE REQUIREMENTS:

➢ Processor	: Any processor above 500 MHz
➢ RAM	: 512MB
➢ Hard Disk	: 10 GB
➢ Input device	: standard Keyboard and Mouse

1.5:SOFTWARE REQUIREMENTS:

➢ Operating system	: Windows 7
➢ Programming language	: Python
➢ Compiler	: Python 3.7.4 , anaconda3 ,pycharm

CHAPTER 2

DATA TYPES:

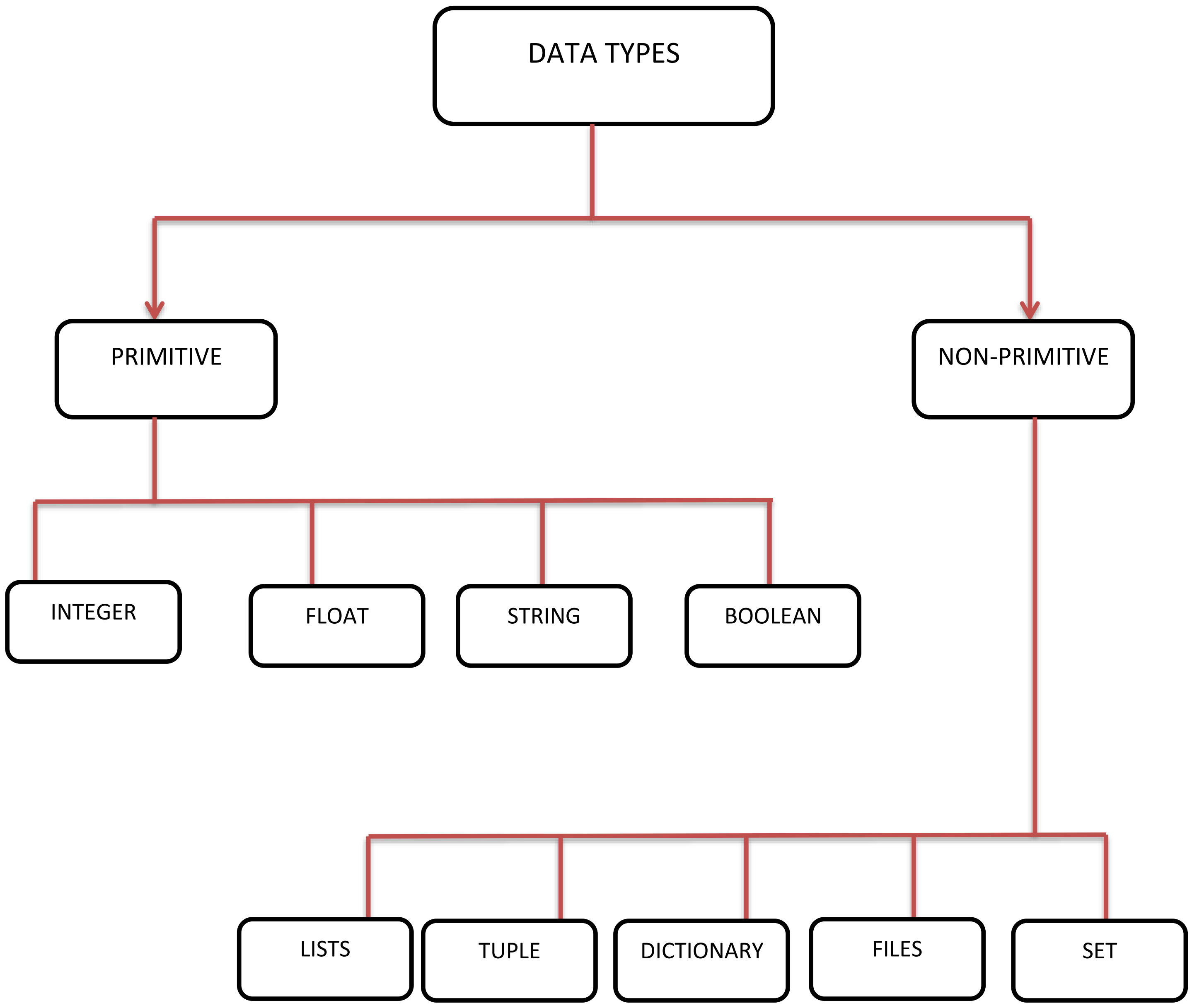


Fig 2.1

Data type are classified into two types :

They are mainly classified based on the storage of data.

1. Primitive data typess: Used to store single values and have no special characteristics

Ex: INTEGER, FLOAT, STRINGS, BOOLEAN

2. Non-primitive data types: derived from primitive data structures, used to store group

Of values with special characteristics.

Ex: lists, dictionary, files, set

1.LISTS:

A list is similar to an array that consists of a group of elements or items. Just like the array, a list can be store elements. But, there is one major difference between an array and a list. An array can store only one type of elements whereas a list can store different types of elements. Hence lists are more versatile and useful than an array.

Creating a List:

Creating a lists is simple by putting different comma-separated values b/w square brackets.

EX: [1,’A’,3]

We can create empty list without any elements by simply writing empty square brackets

Accessing Values in list:

To access values in list, use a squares brackets for slicing along with the index or indices to obtain values available at that index. To view the elements of a list as a whole, we can simply pass the list name to print function.

Positive indexing:

0	**1**	**2**	**3**	**4**	**5**

Negative indexing:

-5	**-4**	**-3**	**-2**	**-1**

METHODS OF LIST:

METHODS	DESCRIPITON
Lst.index(x)	Returns the first occurrence of x in the list.
lst.append(x)	Appends x at the end of the list.
lst.insert(i,x)	Inserts x to the list in the position specified by i.
lst.copy()	Copies all the list elements into a new list and returns it.
lst.extend(lst2)	Appends lst2 to list.
lst.count(x)	Returns number of occurrences of x in the list.
lst.remove(x)	Removes x from the list.
lst.pop()	Removes the ending element from the list.
lst.sort()	Sorts the elements of list into ascending order.
lst.reverse()	Reverses the sequence of elements in the list.
lst.clear()	Deletes all elements from the list.

Nested Lists :

Suppose we want to create a matrix with 3 rows 3 columns, we should create a list with 3 other lists as:

mt = [[1, 22, 3] , [4, 5, 6] , [7, 8, 9]]

Here, „mt‟ is a list that contains 3 lists which are rows of the „mt‟ list. Each row

contains again 3 elements as:

```
    [ [ 1, 22, 3] ,

      [ 4, 5,   6] ,    # second row

      [ 7, 8,   9] ]    # third row
```

List Comprehensions:

List comprehensions represent creation of new lists from an iterable object (like a list, set, tuple, dictionary or range) that satisfy a given condition. List comprehensions contain very compact code usually a single statement that performs the task.

3.DICTIONARY:

A dictionary is represents a group of elements arranged in form of key-values pairs.

The first element is considered as "key‟ and the immediate next element is taken as its "value‟. The keys and the its values is separated by a colons (:). All the key-value pairs in a dictionary are inserted in curly braces { }.

To access the elements of a dictionary, we should not use indexing or slicing. For example, dict[0] or dict[1:3] etc. expressions will give error. To access the value associated with a key, we can mention the key name inside the square braces, as: dict[„Name‟].

If we want to know how many key-value pairs are there in a dictionary, we can use the len( ) function.

We can insert a new key and value pair into existing dictionary this is done by mentioning the name of the key and assigning the value to key.

To Test whether a „key‟ is available in a dictionary or not, we can use „in‟ and „not in‟

operators. These operators return either True or False.

We can use any datatypes for value. For example, a value can be a number, string, list, tuple or another dictionary. But keys should obey the rules:

- Keys should be unique. It means, duplicate keys are not allowed. If we enter same key again, the old key will be overwritten and only the new key will be available.
  Eg:

```
Emp={‘nag’:20, ‘ben’:10,’nag’:30}
Print emp;
#output is {‘nag’:30,’ben’:10}
```

- Keys should be immutable type. For example, we can use a number, string or tuples as keys since they are immutable. We cannot use lists or dictionaries as keys. If they are used as keys, we will get “TypeError‟.

Converting Lists into Dictionary:

When we have two lists, it is possible to convert them into a dictionary. For example, we have two lists containing names of countries and names of their capital cities.

There are two steps involved to convert the lists into a dictionary. The first step is to create a „zip‟ class object by passing the two lists to zip( ) function. The zip( ) function is useful to convert the sequences into a zip class object. The second step is to convert the zip object into a dictionary by using dict( ) function.

Dictionary methods:

Methods	**description**
d.clear()	Removes all key-value pairs from dictionary”d‟.
d2=d.copy()	Copies all elements from„d‟ into a new dictionary d2.
d.fromkeys(s [,v])	Create a new dictionary with keys from sequence”s‟ and Values to”v”
d.get(k [,v])	Returns the value associated with key „k‟. If key is not found, it returns “v‟.
d.items()	Returns an object that contains key-value pairs of”d‟. The pairs are stored as tuples in the object.
d.keys()	Returns a sequence of keys from the dictionary”d‟.
d.values()	Returns a sequence of values from the dictionary”d‟.

d.update(x)	Adds all elements from dictionary “x‟ to”d‟.

5.FILES:

Set of bytes of data which is stored in a place it will provide the permanent storage facility for the user.

Text file : This is a normal file when we open it we can see the plain text file like notepad.**.**

Binary files: Binary files are mostly the **.bin** file. Instead of storing them in a text file we will store then in a binary format (0’s and 1’s).They can store high amount of data compared to text file but they are not easily understand by the user.

Some File Handling Program Examples

- Listing all the files which is present in a dictionary .
- Read the informations from the file and display it on the screen.
- Finding the exact size of the file.
- Create a new file to store information on it.
- Reverse the content of the file and store it
- Copy the content of the file from one file to another.

DIFFERENT MODES OF OPENING A FILE:-

mode	Descripition
r	opens a file in reading mode only.
w	opens or create a file in writing mode only.
a	opens a file in append mode only.
r+	Here opens a file in both reading and writing mode
w+	Here opens a file in both reading and writing mode.
a+	Here opens a file in both reading and writing mode.
Rb	Here opens a binary file for reading mode only.
Wb	opens or create a binary file for writing mode only.
Ab	opens a binary file for append mode only.
rb+	Here opens a binary file in both reading and writing mode.
wb+	Here opens a binary file in both reading and writing mode.
ab+	Here opens a binary file in both reading and writing mode.

BASIC OPERATION ON FILE :

methods	Descripition
open(fn,'w')	function is a string representing a file name.
open(fn, 'r')	Function is a string representing a file name.
open(fn, 'a')	Function is a string representing a file name. Opens an existing file for appending and returns a file handle.
fh.read()	it returns the content of the fh.
fh.readline()	returns the next line in the file associated with the file handle fh.
fh.readlines()	returns a list each element of which is one line of the file associated with the file handle fh.
fh.write(s):	write in string format in fh.
fh.writeLines(S)	S is a sequence of strings. Write each elements of Str to file associated with the file handler .

Modules:

A module is a files containing is Python definitions and statements. The file name the module name with the modulename.py appended. Within a module, the module name (as strings) available the value of the global variable __name__.

from statement:

A module can be contain executable statements as well function definitions. These statements are intended to initialize the module. They are executed only first time the module name is encountered in an statement.

Each module has that's own private symbol tables, which is used the global symbol table by all functions defined in the modules. Thus, an author of module can uses global variables in the module without worrying about the accidental clashes with the user's global variables.

Modules can import other modules. It was customary but not required to places all import statements at the beginnings of a module (or script, for that matter). The imported modules names is placed in the importing module's global symbol tables.

Pickling in Python:

Pickle is used for serializing and de-serializing Python object structures, also called marshalling or flattening. Serialize refer to the process of converting an object in memory to the byte stream that can be stored on disk or sent over a network. Later on, this character stream can then be retrieved and de-serialized back to a Python object. Any object the python can pickled so that it can saved on disk.

Pickle "arrange in series" the object first before write it to file. Pickling is the way to convert the python objects (lists, dicts, etc.) into a character streams. The idea that this character stream contains all the information necessary to reconstruct the object in another python script.

The pickle module implements binary protocols for arranging in series and de-arranging a Python object structure. Pickling which doesnot change it data form into the byte stream, and “unpickling” is the inverse operation, whereby a byte stream (from a binary file or bytes such as object) is converted back into an object hierarchy.

Application:

1) Saving a program state data the disk so that it can carry on where it left off when restarted (persistence)

2) Sending python data over a TCP connection in a multi-core or distributed system (marshalling)

3) Storing python objects in the database.

4) Converting a arbitrary python objects to a string so that it can be used as a dictionary keys (e.g. for caching & memoization).

5) Pickle is the very useful for when you're working with mech learning algorithms, where you want to save them to be able to make new predictions at a later time, without have to rewrites everything or trains the model all over again.

The following types can be pickled:

- None, True, and False.
- integers, floating point numbers, complex numbers.
- strings, bytes, bytearrays.
- tuples, lists, sets, and dictionaries containing only picklable objects.
- functions defined at the top level of a module (using def, not lambda).
- built-in functions defined at the top level of a module.
- classes that are defined at the top level of a module.

Strings:

Strings are the most popular data types in Python. We can create them by enclosing characters in the quotes. Python treats single quotes as same as double quotes. Creating strings is simple as assigning the value to a variables.

s.upper(): This function returns a copy of a string s to uppercase. As strings are immutable, the original string s will remain same.

SPECIAL FUNCTIONS IN PYTHON USED:

1. Import datetime:

In this module it is used to display the time and date displayed on the system in the time of the program run. And it also access the desktop time and date and displayes in the time of excuetion.

2. Import pickle:

In this module it is used to store the data types like lists, tuples, dictionary etc with any change of that form into the string format and in this is it is easy to access and write into. pickle format and it stores data in binary format.

3. Zip():

In this function it is used to append the two lists simultenously Using this function with same indices and add that elements into the one file in the zip format so by this t is used to create a dictionary easily by this zip file by adding the key values into the list and it's corresponding values into the new list and use the zip function to combine this two list and form the dictionary.

CHAPTER 3:

DESIGN GOAL

ASTROSCIENCE is a software which helps to save the multiple data of the people in same place and it display the zodiac sign and one character of the person by taking the name and date of birth as the input.

Firstly it displays the menu of the function to perform and it displays the time and date on top of program execution and asks the user to enter the choice to enter into the further steps.

2.It asks the user to enter the number of inputs to be taken in creation of the new elements and that data is first stored in the list then it zip that files into dictionary and further store that information in the pickle files. Where pickle is the inbuilt module in python.

3. it asks the user to append a new element or to display the result so in the append function the program asks user to enter the number of inputs to be given to the system so by this time it again open the file that previously opened in creation of the new file.

4.display function displays the elements in that files stored in the format of dictionary and load instruction is used to load the information and the dictionary all elements are displayed.

5.search function in this is used to display the output of the person name entered in the search

Function. If the name entered not found it displays the error of the name not found.

If date of birth not matched then the error of the date of birth not matched is displayed

6. At last if user enters 0 choices then the program ends with execution.

After the end of each process it will show 4 options for the user 1.create new file 2. Append 3. Display 4.Search and 0.Exit if we press main menu it will again show he menu part and if we press 0 it will come out from the contact database .

ALGORITHM

```
    Steps:

1. start

2.Display main menu as below:

            * 1.CREATE A NEW FILE

            * 2.APPEND

            * 3.DISPLAY

            * 4.SEARCHING

            * 0.EXIT

      And also display the time and date.

3. Get choice from the user

    choice 1: call function create new list

    choice 2: Call function append

    choice 3: Call function display

    choice 4: Call function search

    choice 0: Call function exit

4.get the output comparing the name and date of birth entered and display.

5.stop.
```

FLOW CHART:

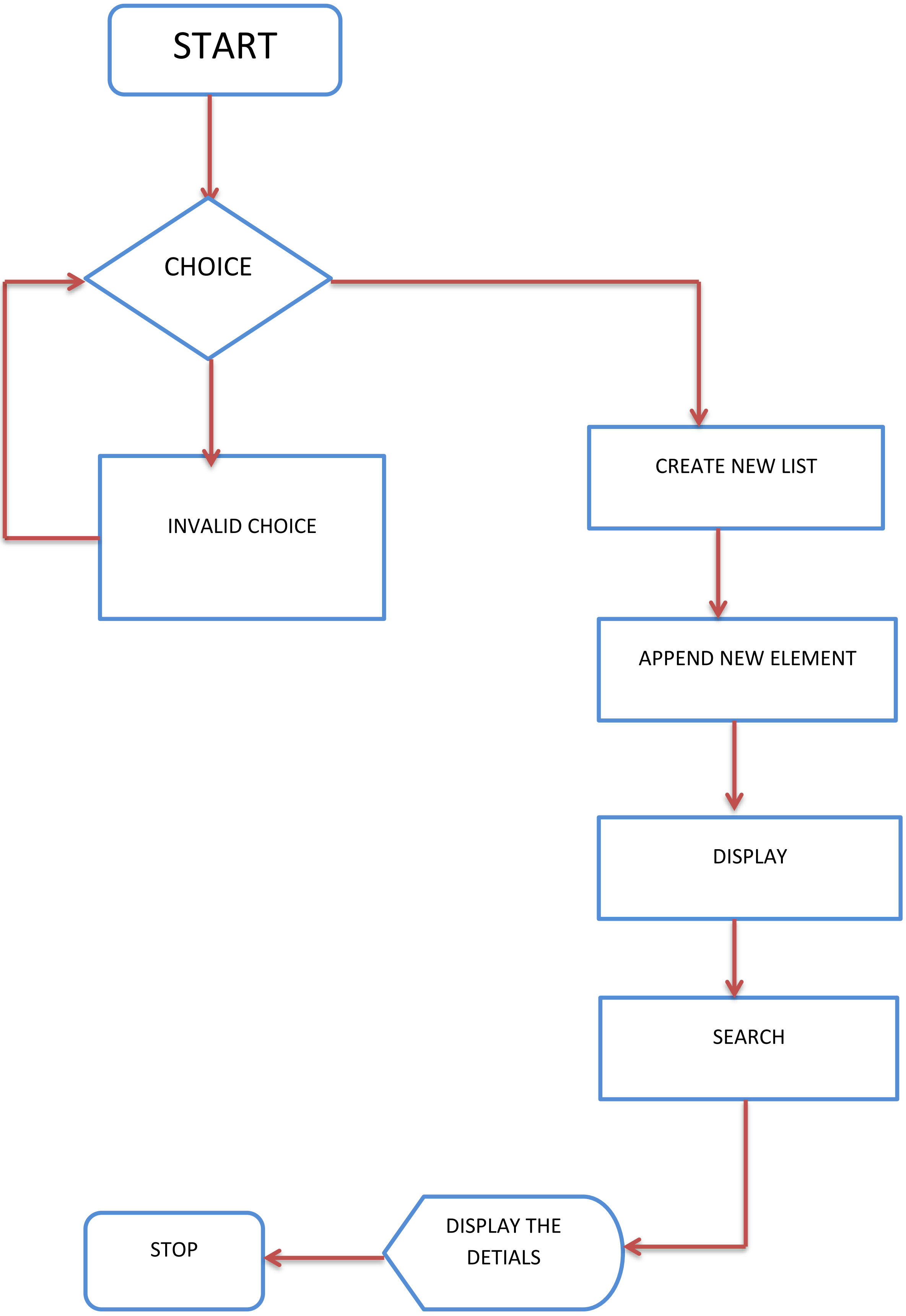


fig 3.1

CHAPTER-4:

IMPLEMENTATION

// before entering into main menu it displays the time and date //

MODULE-1: CREATE A NEW FILE

STEPS:

- display the options for entering the number of tokens.
- Get the data from the user such as name and date of birth.
- Append the name into a list and date of birth into different list.
- Zip the two different lists and convert into a dictionary.
- Open the file and place the dictionary into the file.
- Close the file to perform different operations.
- after performing go back to the main menu.

MODULE 2: APPEND A NEW ELEMENTS

STEPS:

- Again ask the user to enter the new number of tokens.
- Get the data from user such as name and the date of birth
- Append them into a different lists
- Zip the two list and place that in end a previous file.

MOULE 3: DISPLAY

STEPS:

- Open the file in read mode in binary format.
- Display the content of dictionary in the file.
- Close the file.

MODULE 4: SEARCH

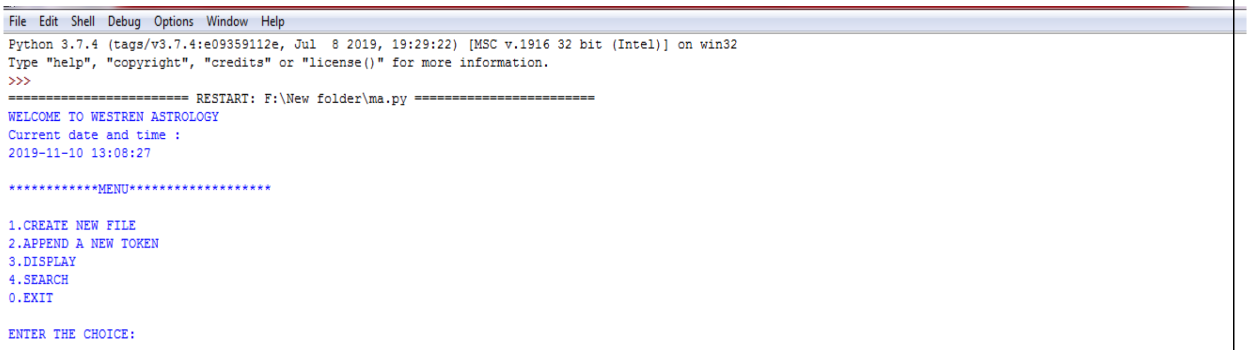
STEPS

- Open the file and display the keys of the dictionary.
- Get the information of user that is to be searched.
- If information found display the user zodiac sign that is present in the other module
- Else display error and close the file.

CHAPTER 5:

RESULTS

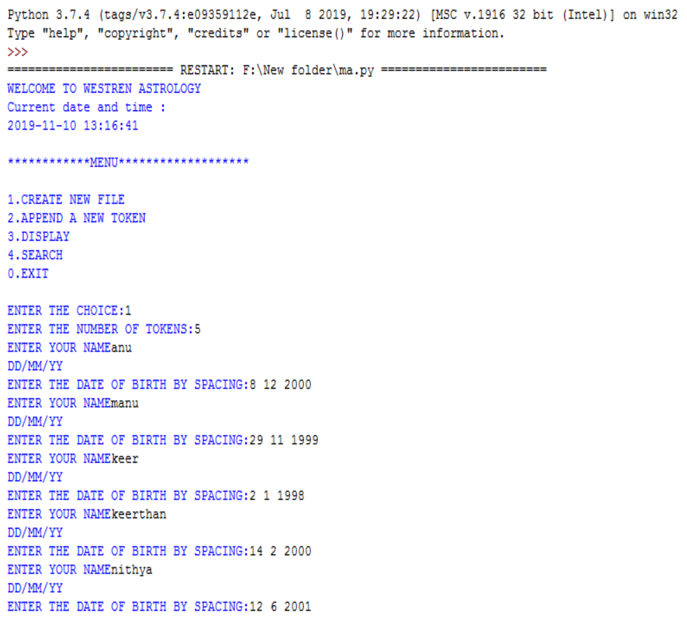
❖ MAIN MENU DISPLAY



```
File  Edit  Shell  Debug  Options  Window  Help
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
======================== RESTART: F:\New folder\ma.py ========================
WELCOME TO WESTREN ASTROLOGY
Current date and time :
2019-11-10 13:08:27

************MENU******************

1.CREATE NEW FILE
2.APPEND A NEW TOKEN
3.DISPLAY
4.SEARCH
0.EXIT

ENTER THE CHOICE:
```

Fig 5.1

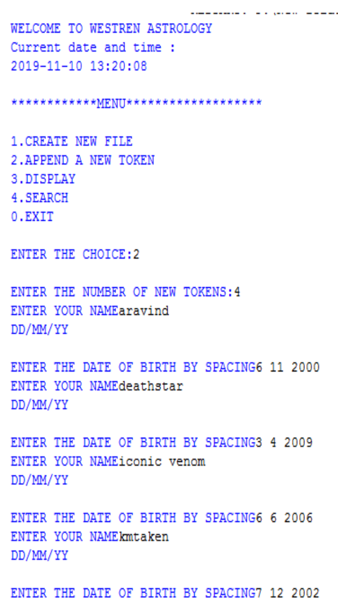
❖ CREATE A NEW FILE



```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
======================== RESTART: F:\New folder\ma.py ========================
WELCOME TO WESTREN ASTROLOGY
Current date and time :
2019-11-10 13:16:41

************MENU******************

1.CREATE NEW FILE
2.APPEND A NEW TOKEN
3.DISPLAY
4.SEARCH
0.EXIT

ENTER THE CHOICE:1
ENTER THE NUMBER OF TOKENS:5
ENTER YOUR NAMEanu
DD/MM/YY
ENTER THE DATE OF BIRTH BY SPACING:8 12 2000
ENTER YOUR NAMEmanu
DD/MM/YY
ENTER THE DATE OF BIRTH BY SPACING:29 11 1999
ENTER YOUR NAMEkeer
DD/MM/YY
ENTER THE DATE OF BIRTH BY SPACING:2 1 1998
ENTER YOUR NAMEkeerthan
DD/MM/YY
ENTER THE DATE OF BIRTH BY SPACING:14 2 2000
ENTER YOUR NAMEnithya
DD/MM/YY
ENTER THE DATE OF BIRTH BY SPACING:12 6 2001
```

Fig 5.2

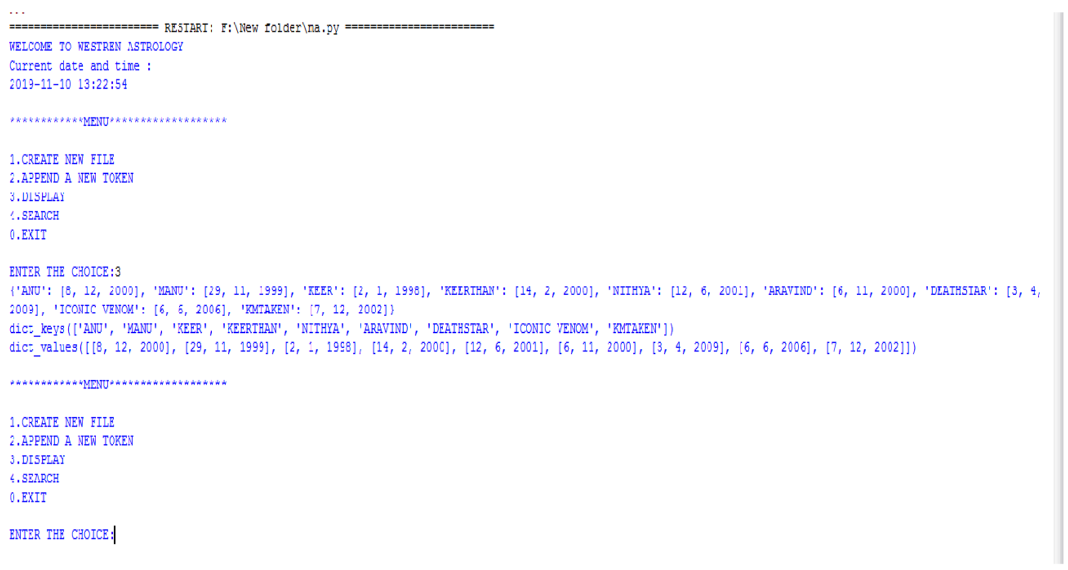
❖ APPEND A NEW DATA



```
WELCOME TO WESTREN ASTROLOGY
Current date and time :
2019-11-10 13:20:08

************MENU******************

1.CREATE NEW FILE
2.APPEND A NEW TOKEN
3.DISPLAY
4.SEARCH
0.EXIT

ENTER THE CHOICE:2

ENTER THE NUMBER OF NEW TOKENS:4
ENTER YOUR NAMEaravind
DD/MM/YY

ENTER THE DATE OF BIRTH BY SPACING6 11 2000
ENTER YOUR NAMEdeathstar
DD/MM/YY

ENTER THE DATE OF BIRTH BY SPACING3 4 2009
ENTER YOUR NAMEiconic venom
DD/MM/YY

ENTER THE DATE OF BIRTH BY SPACING6 6 2006
ENTER YOUR NAMEkmtaken
DD/MM/YY

ENTER THE DATE OF BIRTH BY SPACING7 12 2002
```

Fig 5.3

❖ DISPLAY THE DATA

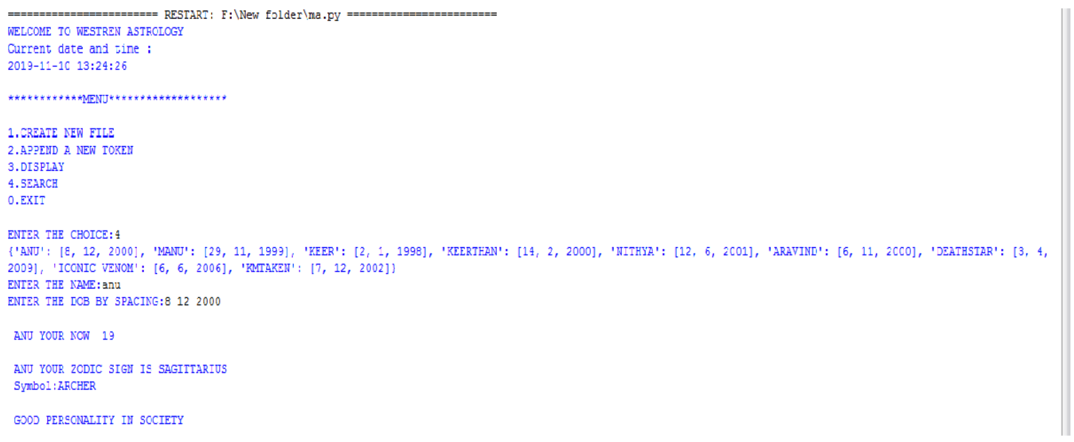


```
...
======================= RESTART: F:\New folder\ma.py =======================
WELCOME TO WESTREN ASTROLOGY
Current date and time :
2019-11-10 13:22:54

************MENU******************

1.CREATE NEW FILE
2.APPEND A NEW TOKEN
3.DISPLAY
4.SEARCH
0.EXIT

ENTER THE CHOICE:3
{'ANU': [8, 12, 2000], 'MANU': [29, 11, 1999], 'KEER': [2, 1, 1998], 'KEERTHAN': [14, 2, 2000], 'NITHYA': [12, 6, 2001], 'ARAVIND': [6, 11, 2000], 'DEATHSTAR': [3, 4,
2009], 'ICONIC VENOM': [6, 6, 2006], 'KMTAKEN': [7, 12, 2002]}
dict_keys(['ANU', 'MANU', 'KEER', 'KEERTHAN', 'NITHYA', 'ARAVIND', 'DEATHSTAR', 'ICONIC VENOM', 'KMTAKEN'])
dict_values([[8, 12, 2000], [29, 11, 1999], [2, 1, 1998], [14, 2, 2000], [12, 6, 2001], [6, 11, 2000], [3, 4, 2009], [6, 6, 2006], [7, 12, 2002]])

************MENU******************

1.CREATE NEW FILE
2.APPEND A NEW TOKEN
3.DISPLAY
4.SEARCH
0.EXIT

ENTER THE CHOICE:
```

Fig 5.4

❖ **SEARCH DATA(**CASE 1)

//SEARCH THE DATA PRESENT IN DATABASE//

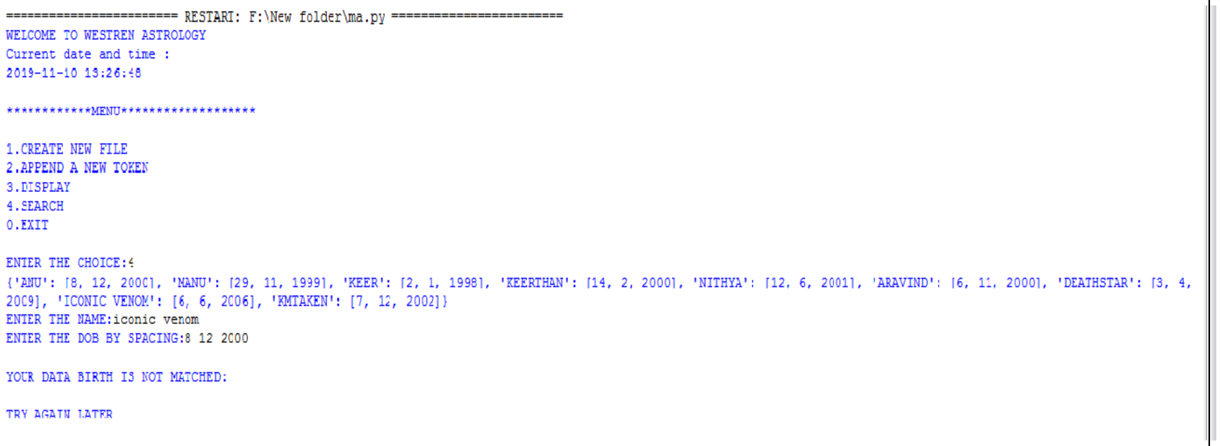


```
======================= RESTART: F:\New folder\ma.py =======================
WELCOME TO WESTREN ASTROLOGY
Current date and time :
2019-11-10 13:24:26

************MENU******************

1.CREATE NEW FILE
2.APPEND A NEW TOKEN
3.DISPLAY
4.SEARCH
0.EXIT

ENTER THE CHOICE:4
{'ANU': [8, 12, 2000], 'MANU': [29, 11, 1999], 'KEER': [2, 1, 1998], 'KEERTHAN': [14, 2, 2000], 'NITHYA': [12, 6, 2001], 'ARAVIND': [6, 11, 2000], 'DEATHSTAR': [3, 4,
2009], 'ICONIC VENOM': [6, 6, 2006], 'KMTAKEN': [7, 12, 2002]}
ENTER THE NAME:anu
ENTER THE DOB BY SPACING:8 12 2000

 ANU YOUR NOW  19

 ANU YOUR ZODIC SIGN IS SAGITTARIUS
 Symbol:ARCHER

 GOOD PERSONALITY IN SOCIETY
```

Fig 5.5

❖ **SEARCH DATA(**CASE 2)

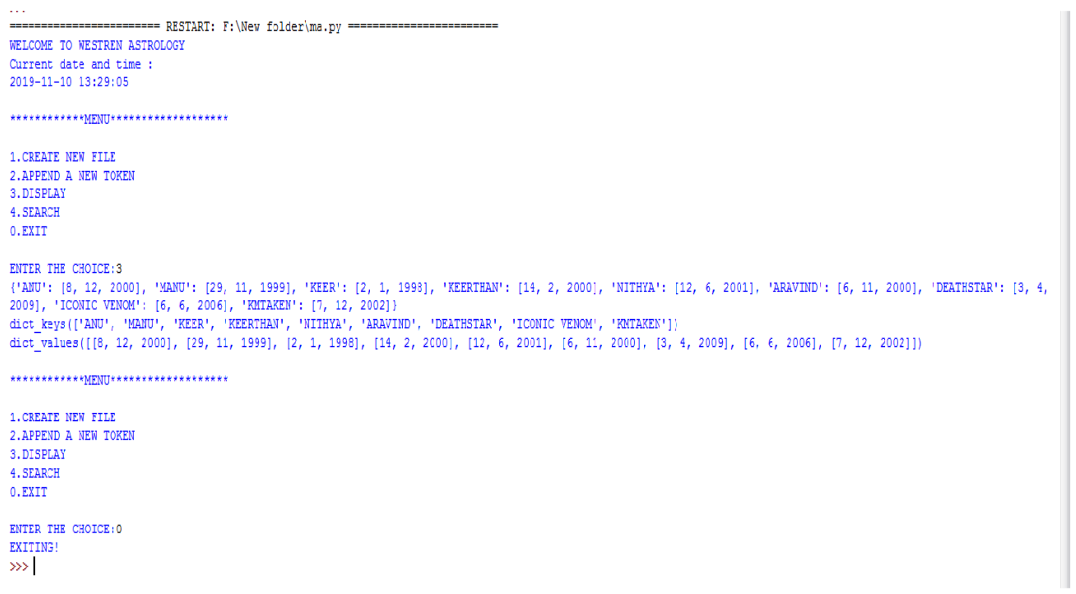
// SEARCH DATA NOT IN DATABASE//



```
======================= RESTART: F:\New folder\ma.py =======================
WELCOME TO WESTREN ASTROLOGY
Current date and time :
2019-11-10 13:26:48

************MENU******************

1.CREATE NEW FILE
2.APPEND A NEW TOKEN
3.DISPLAY
4.SEARCH
0.EXIT

ENTER THE CHOICE:4
{'ANU': [8, 12, 2000], 'MANU': [29, 11, 1999], 'KEER': [2, 1, 1998], 'KEERTHAN': [14, 2, 2000], 'NITHYA': [12, 6, 2001], 'ARAVIND': [6, 11, 2000], 'DEATHSTAR': [3, 4,
2009], 'ICONIC VENOM': [6, 6, 2006], 'KMTAKEN': [7, 12, 2002]}
ENTER THE NAME:iconic venom
ENTER THE DOB BY SPACING:8 12 2000

YOUR DATA BIRTH IS NOT MATCHED:

TRY AGAIN LATER
```

Fig 5.6

❖ EXITING



```
...
======================== RESTART: F:\New folder\ma.py ========================
WELCOME TO WESTREN ASTROLOGY
Current date and time :
2019-11-10 13:29:05

************MENU*******************

1.CREATE NEW FILE
2.APPEND A NEW TOKEN
3.DISPLAY
4.SEARCH
0.EXIT

ENTER THE CHOICE:3
{'ANU': [8, 12, 2000], 'MANU': [29, 11, 1999], 'KEER': [2, 1, 1998], 'KEERTHAN': [14, 2, 2000], 'NITHYA': [12, 6, 2001], 'ARAVIND': [6, 11, 2000], 'DEATHSTAR': [3, 4,
2009], 'ICONIC VENOM': [6, 6, 2006], 'KMTAKEN': [7, 12, 2002]}
dict_keys(['ANU', 'MANU', 'KEER', 'KEERTHAN', 'NITHYA', 'ARAVIND', 'DEATHSTAR', 'ICONIC VENOM', 'KMTAKEN'])
dict_values([[8, 12, 2000], [29, 11, 1999], [2, 1, 1998], [14, 2, 2000], [12, 6, 2001], [6, 11, 2000], [3, 4, 2009], [6, 6, 2006], [7, 12, 2002]])

************MENU*******************

1.CREATE NEW FILE
2.APPEND A NEW TOKEN
3.DISPLAY
4.SEARCH
0.EXIT

ENTER THE CHOICE:0
EXITING!
>>>
```

Fig 5.7

CHAPTER 6:

CONCLUSION

This project **ASTROSCIENCE** mainly deals with the information of the people this was developed to get the information instantly. This idea is very useful in the condition like in maintaining the record of certain people and where we can handle huge amount of data in same place .

The aim of this project was to deal with the storing of data’s , to implement new ideas like to modify the existing data(manipulating or rewriting), Handling of files ,working with looping statements and usage of the new inbuilt modules etc by using the python programming language .

REFERENCES:

https://www.programiz.com/python-programming/pickle

https://www.geeksforgeeks.org/python-dictionary

https://stackoverflow.com/questions