

Koneru Lakshmaiah Education Foundation
(Deemed to be University)

FRESHMAN ENGINEERING DEPARTMENT

A Project Based Lab Report

On

CONTEMPORARY PERIODIC TABLE

SUBMITTED BY:

I.D NUMBER NAME

2300030652 SRIRAM PRAVALLIKA

UNDER THE GUIDANCE OF

DR.E.SRIDEVI

Associate Professor,CSE.



KL UNIVERSITY
Green fields, Vaddeswaram – 522 502
Guntur Dt., AP, India.

DEPARTMENT OF BASIC ENGINEERING SCIENCES-1



CERTIFICATE

This is to certify that the project based laboratory report entitled “CONTEMPORARAY PERIODIC TABLE IN C” submitted by Ms. SRIRAM PRAVALLIKA bearing Regd.No.2300030652 to the Department of Basic Engineering Sciences-1, KL University in partial fulfillment of the requirements for the completion of a project based Laboratory in “Computational Thinking for Structural Design ”course in I B Tech I Semester, is a bonafide record of the work carried out by her under my supervision during the academic year 2023 – 2024.

PROJECT SUPERVISOR

Dr. E. SRIDEVI

HEAD OF THE DEPARTMENT

DR.D. HARITHA

ACKNOWLEDGEMENTS

It is great pleasure for me to express my gratitude to our honorable President Sri. Koneru Satyanarayana, for giving the opportunity and platform with facilities in accomplishing the project based laboratory report.

I express the sincere gratitude to our Director Dr. A.Jagadeesh for his administration towards our academic growth.

I express sincere gratitude to HOD-BES-1 Dr. D.Haritha for her leadership and constant motivation provided in successful completion of our academic semester. I record it as my privilege to deeply thank for providing us the efficient faculty and facilities to make our ideas into reality.

I express my sincere thanks to our project supervisor DR.SRI DEVI for her novel association of ideas, encouragement, appreciation and intellectual zeal which motivated us to venture this project successfully.

Finally, it is pleased to acknowledge the indebtedness to all those who devoted themselves directly or indirectly to make this project report success.

Sriram Pravallika

Project Associates...

ABSTRACT

This project aims to create a dynamic and user-friendly application in C language for representing a temporary periodic table. The periodic table is a fundamental tool in chemistry, showcasing the properties and relationships of chemical elements. This program will provide a flexible and extensible platform for users to interact with and explore elemental data.

INDEX

S.NO	TITLE	PAGE NO
1	Introduction	1
2	Aim of the Project	7-8
2.1	Advantages & Disadvantages	8-9
2.2	Future Implementation	9-10
3	Software & Hardware Details	10
4	Flow Chart	11
5	Algorithm	12
6	Implementation	12-17
7	Integration and System Testing	17,18
8	Conclusion	20

INTRODUCTION

Concepts:

Structures, Functions, Files, Switch case

Function:

A function is an individual component to reduce the complexity of code. Functions are of four types. Functions are also called as Modular Programming. Functions consists of declaration, function call and function definition.

This project will help you to understand file handling in C i.e. creating a file and accessing the stored data in the file, modifying and removing the stored data. It will also help you to understand the use of functions as well as different parameters of C programming language.

Storage of Element Information :

In the project, you can add any new element with its name, symbol, atomic number, atomic weight and its some important properties. When new element information is to be added to this Modern Periodic Table, you have to enter 1 in the main menu and input information in given format. These information are stored in file created on the hard disk of computer by program itself.

AIM OF THE PROJECT:

The aim of a temporary periodic table program in C language is to create a software application that allows users to interact with and explore the properties of chemical elements in a dynamic and flexible manner. The key objectives of such a program can include:

Data Representation:

Create a data structure to represent the periodic table dynamically. This involves storing information about each chemical element, such as atomic number, symbol, name, and other relevant properties.

User Interaction:

Design an interactive user interface that enables users to navigate and explore the periodic table. Users should be able to select elements, view detailed information, and interact with the data in an intuitive manner.

Dynamic Updates:

Implement mechanisms to dynamically update and modify the data within the periodic table. This could involve adding new elements, updating properties, or adjusting the organization of the table.

Customization Options:

Allow users to customize the display of the periodic table based on their preferences. This may include sorting options, filtering elements based on specific criteria, and highlighting certain properties.

Educational Features:

Incorporate educational features to enhance the learning experience. This could involve providing additional information about each element, tooltips with relevant details, and possibly interactive educational modules.

Cross-Platform Compatibility:

Ensure that the program is compatible with different operating systems, making it accessible to a wider audience. This may involve writing the code in a way that is portable and can be compiled on various platforms.

Localization:

Support multiple languages to cater to a global audience. This makes the program accessible and user-friendly for individuals who speak different languages.

Ease of Data Management:

Develop efficient algorithms and data management techniques to

handle the dynamic nature of the periodic table. This includes handling updates, deletions, and additions to the data.

Educational and Scientific Utility:

Provide a tool that is not only user-friendly but also serves educational and scientific purposes. This can benefit students, educators, and researchers who want to explore and understand the properties of chemical elements.

Scalability:

Design the program to be scalable, accommodating potential future updates to the periodic table or additional features. This ensures that the program remains relevant and useful over time.

ADVANTAGES

Temporary variables are often used to store intermediate values during computations or to simplify complex expressions. Here are some advantages:

Readability:

Using temporary variables can make your code more readable by breaking down complex expressions into smaller, more manageable parts. This can make it easier for others (or yourself) to understand the logic behind the code.

Debugging:

When debugging, it's often helpful to inspect the intermediate values of variables. Temporary variables can be used to store these values and make it easier to identify and fix issues in your code.

Code Optimization:

Compilers can sometimes optimize code better when temporary variables are used. The use of temporary variables can help the compiler to optimize the code for better performance.

Reuse of Values:

If a value is used multiple times in a calculation, storing it in a temporary variable can avoid redundant computations, improving efficiency.

DISADVANTAGES

Limited Accuracy:

Storing the properties of chemical elements in a program may require approximations or simplifications. The periodic table is a sophisticated representation, and any simplification may lead to loss of accuracy.

Maintenance Complexity:

If the periodic table data changes or needs updates, modifying the program may be cumbersome. This could be an issue if the data is not easily modifiable or if the program needs to be redeployed for each update.

Memory Usage:

Representing a large amount of data for all the elements in the periodic table might consume a significant amount of memory. This can be a concern if your program needs to run on resource-constrained devices.

Processing Overhead:

If your program frequently accesses or manipulates the periodic table data, there could be a performance overhead. Efficient algorithms and data structures need to be implemented to minimize processing time.

Compatibility Issues:

Depending on the purpose of your program, the periodic table data might need to be compatible with other systems or standards. Ensuring compatibility could be a challenge, especially if there are changes in data formats or standards.

Data Integrity:

Ensuring the integrity of the periodic table data is crucial. In a dynamic program environment, errors or corruption in the data could lead to inaccurate results or unexpected behavior.

Localization:

If your program is intended for a global audience, you might need to consider localization issues, such as displaying the periodic table in different languages or adapting to different regional conventions.

Future Enhancements:-

- Dynamic Data Loading
- Database Integration
- Data Validation and Error Handling
-

SYSTEM REQUIREMENTS

➤ SOFTWARE REQUIREMENTS:

The major software requirements of the project are as follows:

Language : C

Operating System: Windows Xp or later.

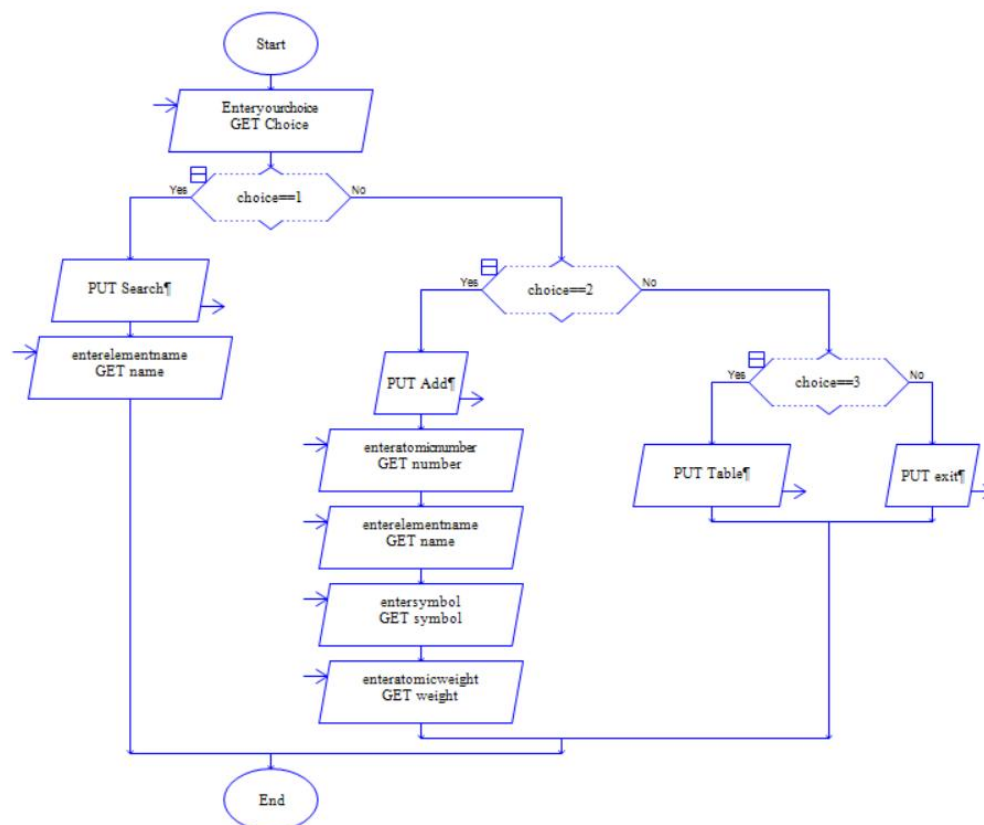
➤ HARDWARE REQUIREMENTS:

The hardware requirements that map towards the software are as follows:

RAM : 4GB/8GB

Processor : intel i3, i5, i7

FLOW CHART



ALGORITHM

Step 1: Start

Step 2: Enter your choice

Step 3: if(choice==1)

Step 4: Controller enters into search function.

Step 5: elseif(choice==2)

Step 6: Controller enters into add function

Step 7: elseif(choice==3)

Step 8: Controller enters into display function

Step 9: elseif(choice>3)

Step 10: Controller comes out of the loop.

Step 11: Create a File for storing and printing the Element details.

Step 12: When controller is in add function.

Step 13: Enter element atomic number, name, atomic weight, symbol.

Step 14: When the controller is in search function ,enter your element to search if the element exists in prints Search successful else it prints Search unsuccessful ,try again.

Step 15: When controller is in display function it displays the periodic table.

Step 16: Stop

IMPLEMENTATION

```
#include <stdio.h>

void searchelement();

void add();

void display();

struct Element
{
    int enumber;
    char esymbol[3];
    char ename[20];
    float emass;
};

int main()
{
    int choice;
    const char *filename = "element.txt"
    while (1)
    {
        printf("1. Search element\n2. Add element\n3. Display periodic
table\n");

        printf("Enter your choice:\n");
        scanf("%d", &choice);
        if (choice == 1)
            searchelement();
```

```

    else if (choice == 2)
        add();
    else if (choice == 3)
        display(filename);
    else if (choice > 3)
        printf("Enter correct choice\n");
    }
return 0;
}
void add()
{
    struct Element e;
    printf("Enter atomic number:");
    scanf("%d", &e.enumer);
    printf("Enter name of the element:");
    scanf("%s", e.ename);
    printf("Enter symbol:");
    scanf("%s", e.esymbol);
    printf("Enter atomic weight:");
    scanf("%f", &e.emass);
    FILE *file = fopen("element.txt", "a");
    if (file == NULL)
    {
        printf("Error opening file.\n");
    }
}

```

```

        return;
    }
    fprintf(file, "%d %s %s %.3f\n", e.enumer, e.esymbol, e.ename, e.emass);
    fclose(file);
    printf("\nElement added successfully\n");
}
void searchelement()
{
    struct Element e;
    int k = 0;
    char name[20];
    FILE *file = fopen("element.txt", "r");

    if (file == NULL)
    {
        printf("Error opening file.\n");
        return;
    }
    printf("Enter the element to search: ");
    scanf("%s", name);

    while (fscanf(file, "%d %2s %19s %f", &e.enumer, e.esymbol, e.ename,
&e.emass) == 4)
    {

```

```

        if (strcmp(name, e.ename) == 0)
        {
            k = 1;
            printf("Search Successful\n");
            break;
        }
    }
    if (k != 1)
        printf("Search Unsuccessful\n");

    fclose(file);
}

void display(const char *filename)
{
    FILE *file = fopen(filename, "r");
    if (file == NULL)
    {
        printf("Error opening file.\n");
        return;
    }
    struct Element element;
    printf("Periodic Table\n");
    printf("-----\n");

```



```

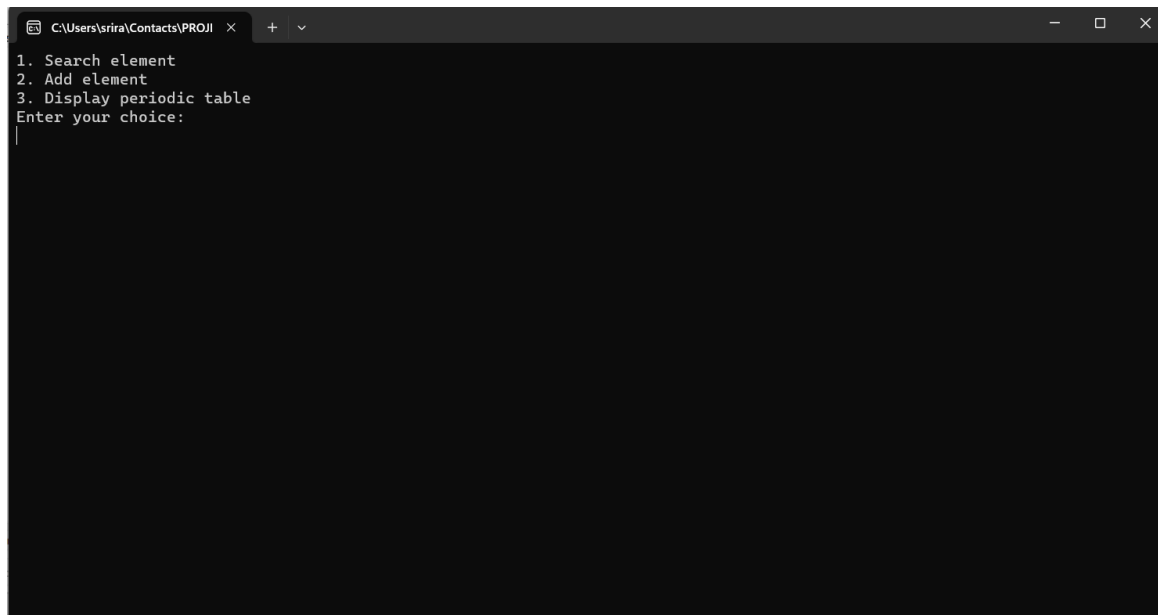
printf("| %-3s | %-2s | %-20s | %-10s |\n", "No.", "Sym", "Name", "Mass");
printf("-----\n");

while (fscanf(file, "%d %2s %19s %f", &element.number,
element.esymbol, element.ename, &element.emass) == 4)
{
    printf("| %-3d | %-2s | %-20s | %-10.3f |\n", element.number,
element.esymbol, element.ename, element.emass);
}
printf("-----\n");
fclose(file);
}

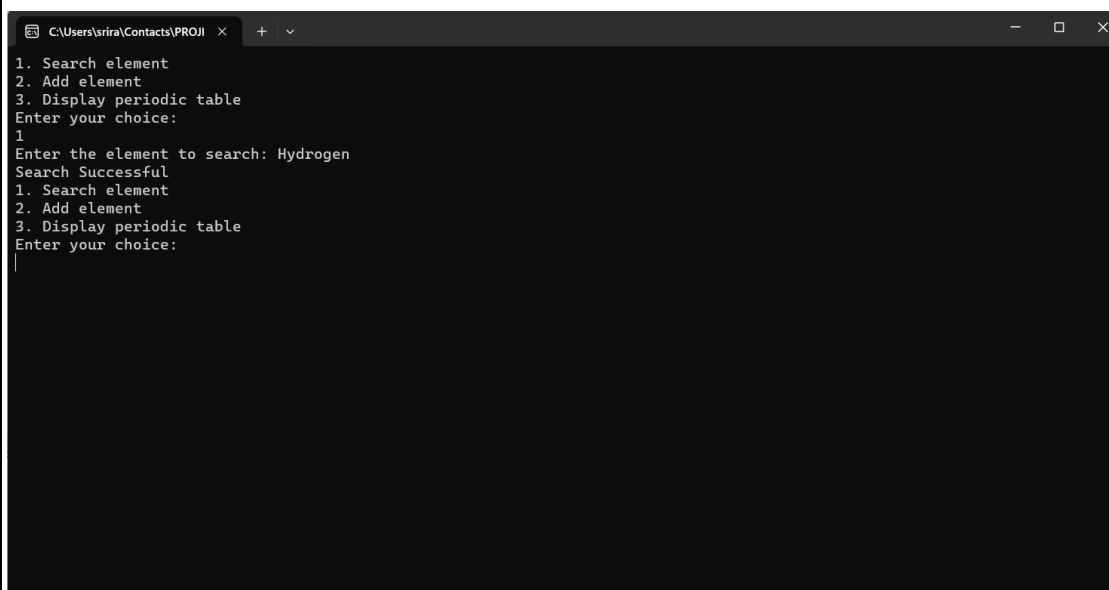
```

INTEGRATION AND SYSTEM TESTING

OUTPUTS



```
C:\Users\sriira\Contacts\PROJI x + v
1. Search element
2. Add element
3. Display periodic table
Enter your choice:
|
```



```
C:\Users\sriira\Contacts\PROJI x + v
1. Search element
2. Add element
3. Display periodic table
Enter your choice:
1
Enter the element to search: Hydrogen
Search Successful
1. Search element
2. Add element
3. Display periodic table
Enter your choice:
|
```

```
C:\Users\sriira\Contacts\PROJ1 x + v
1. Search element
2. Add element
3. Display periodic table
Enter your choice:
2
Enter atomic number:16
Enter name of the element:Sulphur
Enter symbol:S
Enter atomic weight:32

Element added successfully
1. Search element
2. Add element
3. Display periodic table
Enter your choice:
|
```

```
C:\Users\sriira\Contacts\PROJ1 x + v
1. Search element
2. Add element
3. Display periodic table
Enter your choice:
3
Periodic Table
-----
| No. | Sym | Name           | Mass  |
-----|-----|-----|-----|
| 1   | H   | Hydrogen       | 1.008 |
| 2   | He  | Helium         | 4.000 |
| 3   | Li  | Lithium        | 6.000 |
| 4   | Be  | Beryllium      | 9.000 |
| 5   | B   | Boron          | 10.000|
| 6   | C   | carbon         | 12.000|
| 7   | N   | Nitrogen       | 14.000|
| 1   | H   | Hydrogen       | 1.008 |
| 8   | O   | Oxygen         | 16.000|
| 9   | f   | Flourine       | 18.000|
| 10  | Ne  | Neon           | 20.000|
| 11  | Na  | Sodium        | 23.000|
| 12  | Mg  | Magnesium      | 24.000|
| 13  | Al  | Aluminium      | 26.000|
| 14  | Si  | Silicon        | 28.000|
| 15  | P   | Phosphorous    | 30.000|
| 16  | S   | Sulphur        | 32.000|
-----
1. Search element
2. Add element
3. Display periodic table
```

CONCLUSION

In conclusion, the contemporary periodic table, organized based on atomic number and electronic configuration, stands as a fundamental tool in understanding the properties and relationships of elements.