**JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY**



**MINI PROJECT - DATA STRUCTURE**

**PERIODIC TABLE**

GAURAV (14103215)

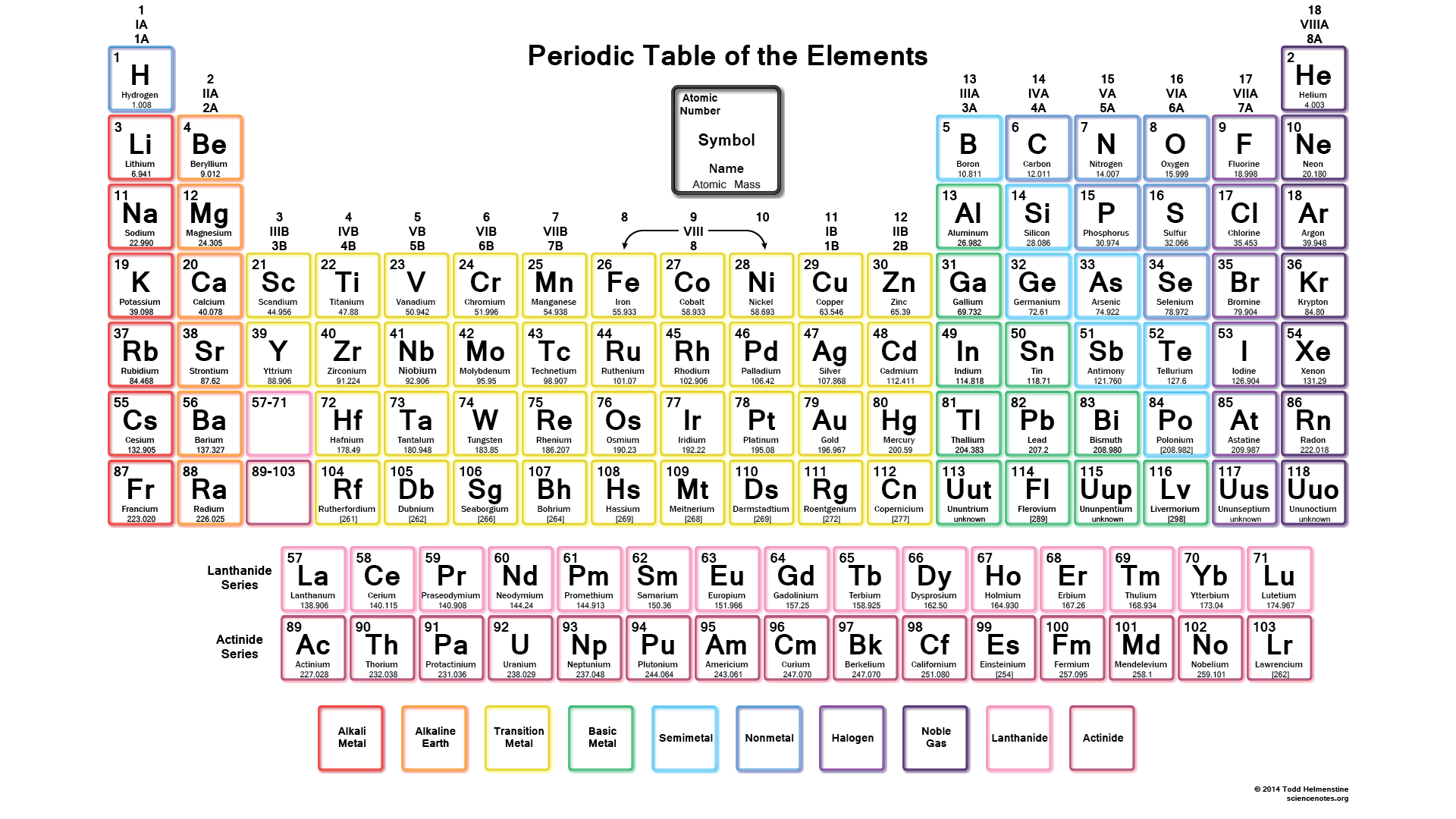
SHUBHANK (14103221)

ANKIT (14103228)

PRAGESH (14103239)

BATCH – B7

**INTRODUCTION**



Firstly, as we compile and run our program, user will get a glimpse of Modern periodic table.

Secondly, we will ask the user to enter the name of the element he/she want’s to know about

Eg.Li

Atomic mass = 7

Atomic no. =3

Symbol = Li

Group=1st

Period=2nd

Last but not the least there will be a little quiz if the user wants to know test himself/herself.

**CONCEPTS USED IN PROJECT - :**

* Link list
* Multi link list

Link list and multi link list are the primary data structures used in our project. As they are being used to store basic information such as atomic mass ,atomic number , symbol and various other parameters.

* Graphics

Displaying the periodic table when the program runs initially and also in displaying the electronic configuration of the elements depicting the total no. of shells and total no. of electrons in each shell.

* File Handling

File handling has being used in storing and retrieving the information of various elements of periodic table.

* Function(recursion)

Various functions is being used to design the outlook of our project. Also , wide use of recursive function in making tree is being used in make-up of our project.

* TREE

Binary search Tree has been used to store various elements of the periodic table, to display them in alphabetical order and for efficient searching of elements.

**PROBLEM STATEMENT**

The main theme of the project is to store,search & display the information of various elements by using a multi linkedlist and a binary search tree . Since,the average time complexity of searching in binary search tree is O(logn) and therefore it is more efficient in searching elements than the multi linked list whose average time complexity of searching is

BRIEF DESCRIPTION OF MAIN FUCTIONS

1.void insertnode(ll \*l,element e);

This function takes two parameter , a link list and a structure which stores the information of a particular element retrived from file. This function creates a node , allocates memory for this node and stores the information fetched by the function .This function creates a multi- linked list having two pointers one right pointer which links elements of period and one down pointer which links elements of group.Since lanthanoids and actinoids belong to the same group and period respectively this function also links these elements with the help of an in pointer.

2.SEARCHING FUNCTIONS

There are various searching functions used in the program such as

void searchbyatomic(ll \*l,atomic);

void searchbysymbol(ll \*l,symbol);

void searchbyname(ll \*l,name);

void searchby\_gp\_pd(ll \*l,group,period);

since it is the main requirement of the project.To search any element by various parameters we require that particular parameter and address of starting element of the link list. These functions first search the required element in the first period and if the element is not found , it searches the element in the subsequent periods and repeats the same process.

**3.tnode \* inserttnode(tnode \*root,element e);**

This function takes two parameters,one is the address of root of the tree and other is the

structure which stores the information of a particular element retrived from file.This function creates a tnode , allocates memory for this tnode and stores the information fetched by the function. Since it is a binary search tree,all elements which are having names which are lexicographically less the the root’s name are placed on the left side of the root and whereas all the other elements are placed on the right side side of the root.

Storing data in such a format helps us to search elements quickly as the average time complexity of searching in binary search tree in O(logn) as compared to linked list average time complexity of searching i.e. O(n).We can also display the element’s information sorted in alphabetical order via inorder traversal.

**Algorithm of Main Functions**

1.void insertnode(ll \*l,element e);

1.1 Declare 3 variables temp,temp1 and temp2 of type node \*.

1.2 temp1=temp2=l->start;

1.3 We create a node by allocating memory and storing elements retrived from file and we point temp towards the node.

temp=createnode(e);

1.4 IF(l->start==NULL)

THEN l->start=temp;

ELSE

{

IF(temp->period==temp1->period)

THEN temp1->next=temp;

ELSE IF(temp->group==temp1->group)

{

WHILE(temp1->down!=NULL)

temp1=temp1->down;

temp1->down=temp;

}

ELSE

{

WHILE(temp1->period!=temp->period)

temp1=temp1->down;

WHILE(temp1->next!=NULL)

temp1=temp1->next;

IF(((temp->atomic>=58)&&(temp->atomic)<=72)||(temp->atomic>=90)&&(temp->atomic)<=103)

{

IF((temp->atomic==72)||(temp->atomic==103))

temp1->next=temp;

WHILE(temp1->in!=NULL)

temp1=temp1->in;

temp1->in=temp;

}

ELSE

temp1->next=temp;

WHILE(temp2->period!=temp->period-1)

temp2=temp2->down;

WHILE(temp2->next!=NULL)

{

temp2=temp2->next;

IF(temp2->group==temp->group)

{

IF((temp->atomic>=58)&&(temp->atomic<=72)||(temp- >atomic>=90)&&(temp->atomic<=103))

BREAK;

temp2->down=temp;

BREAK;

}

}

}

}

2. void searchby\_gp\_pd(ll \*l,group,period);

2.1 Declare a variable temp1 of type node \*.

2.2 temp1=l->start;

2.3 Move temp=temp->down till we find temp->period==period

2.3.1 IF condition is satisfied Move temp=temp->right till we find

temp->group==group

2.3.2 Display the information of the element found.

2.4 IF Element is NOT found

THEN print “Element not found” and BREAK

3.tnode \* inserttnode(tnode \*root,element e);

3.1 IF(root==NULL)

THEN tnode \*temp=createtnode(e);

root=temp;

3.2 ELSE

{

3.2.1 IF(strcmp(root->name,e.name)>0)

root->left=inserttnode(root->left,e);

ELSE

root->right=inserttnode(root->right,e);

}

3.3 return root;

3.4 END OF FUNCTION.