



Project Report

(CSE1021) Introduction to Problem Solving and Programming

Slot-A14+D11+D12, BL2025260100809

Fall Semester: 2025-2026

Name of Student: Akshat Rajput

Register Number: 25BCE10874

Introduction-

“**ChemBro**” is a python - based educational application designed to help visualize electron shell diagrams and access extensive information about chemical elements. It also contained a detailed dataset of elements including physical, chemical, and configuration details.

The system uses: -

Dictionary- Collection of data stored in key-value pair. Instead of using numbered index to retrieve data we use key to find associated value.

Its Mutable

It is enclosed in { }.

Ex- {“name”: “Akshat”, “Age”:19}

List- An ordered mutable collection of elements. You can add, remove, or change items within a list after it is created.

Syntax- Square Bracket []

Example- [“apple”, “banana”, 5, 3.14]

String- A sequence of character used to represent text. String is immutable, meaning once created they cannot be changed.

Syntax- Enclosed in single ‘ ’ or double “ ” quotes.

Example- “Hello World”

Tuple- Like list a tuple is an ordered sequence but its immutable.

Syntax- Enclosed in parentheses ().

Example-(19, 15, “Akshat”)

Functions-It is a reusable block of code that is designed to perform specific task. Instead of typing same code repeatedly we can define a function and call it whenever it’s needed with manipulation in parameter or argument

Turtle- In Build python library that can be used to draw figures as programmed.

Tkinter- In Build python library used for creating user interface (GUI).

PyAutoGUI – Python module used for GUI automation. It allows your python scripts to programmatically control the mouse and keyboard, enabling the computer to perform tasks just as a human user would.

Problem Statement and Objective-

Students struggle to understand the structure of atoms and electron configurations as existing tools are complex to use.

So, this is a simple, intuitive tool that automatically displays electron shell diagrams and retrieves detailed element information from a structured dataset.

Functional Requirement-

- 1- Input Atomic Number
User enter atomic number on terminal
- 2- Calculate Electron Distribution
Use Sequence as 1s,2s,2p,3s to calculated electron distribution.
- 3- Generate Electron Shell Diagram
The Program draw electron shell diagram using turtle first it draws Nucleus with Turtle than it draws orbit (K,L,M,N) than it generates Electron on orbit.
- 4- Retrieve Element Data
Program fetch name, appearance, atomic mass, shells, electronic Configuration and other details from dictionary.
- 5- According to user input in display the output to user like they want to see element details or electron shell diagram.

Non- Functional Requirements-

- 1- Usability- The text interface should be simple, prompting user for numeric Inputs.
- 2- Performance- Program should be able to render diagram in very less time.
- 3- Reliability- The program is fully on python and standard libraries like turtle, time, tkinter and some 3rd party library like pyautogui, keyboard.
- 4- Accuracy- To provide user accurate data program uses all information scaped from verified source.
- 5- Security- Program runs locally to prevent any type of cyber treat.

System Architecture-

The program uses this architect to produce output: -

- **Data Layer-** Program uses python dictionary element to produce output.
- **Logic Layer-**
 - Program uses diag() function to calculate electron distribution of element.
 - Program uses drawing () function to draw electron shell diagram.
 - It uses dictionary retrieval to produce all details about element and

- **Presentation Layer-**

- Program ask input to what user like only details about element Or Electron Shell diagram.

```
print("What you want to print")
print("Press 1 to print All detail about Element")
print("Press 2 to print paricular detail about Element")
print("Press 3 to draw Electron Shell Diagram of given Atomic No.")
n=int(input("Enter value"))
if n==1:
    z = int(input("Enter Atomic No.:"))
    AllDetails()

elif n==2:
    z = int(input("Enter Atomic No.:"))
    print("What do you want to print")
    all=["Name", "Appearance", "Atomic Mass", "Boil", "Category", "Density", "Discover",
        "Group", "Phase", "Summary", "Symbol", "Electron confriguration"]
    print(all)

    Particular_Detail()
elif n==3:
    drawing()
```

Design Decision & Rationale-

- **Data Storage-**

All data is stored in a nested dictionary {Atomic No.: {Key: Value}}

Ex-

```
5:{"Name": "Boron",
   "Appearance": "black-brown",
   "Atomic Mass": 10.81,
   "Boil": 4200,
   "Category": "metalloid",
   "Density": 2.08,
   "Discovered by": "Joseph Louis Gay-Lussac",
   "Melt": 2349,
   "Molar heat": 11.087,
   "Named by": "null",
   "Number": 5,
   "Period": 2,
   "Group": 13,
   "Phase": "Solid",
   "spectral_img": "null",
   "Summary": "Boron is a metalloid chemical element with Symbol B and",
   "Symbol": "B",
   "Shells": [
       2,
       3
   ],
   "Electronic configuration": "1s2 2s2 2p1",
   "Electron configuration semantic": "[He] 2s2 2p1",
   "Block": "p"},
```

- **Visualisation Library –**

- Program uses turtle library to draw electron shell diagram.

- **Input Automation –**

Used pyautogui to maximize window for better visibility.

Implementation Details-

- **Library-** time, tkinter, turtle, pyautogui, keyboard
- **File Contain-**
 - Element List
 - Atomic number
 - Shell Sequence
 - Electron Distribution algorithm
- Runs the function drawing () which triggers shell calculation and rendering.

Screenshots / Results-

- Input screen-

```
What you want to print
Press 1 to print All detail about Element
Press 2 to print paricular detail about Element
Press 3 to draw Electron Shell Diagram of given Atomic No.
Enter value|
```

- Element details displayed-

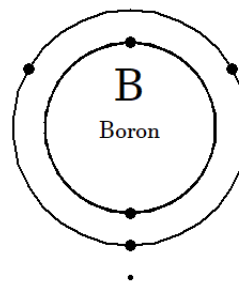
```
= RESTART: D:\chempro.py
What you want to print
Press 1 to print All detail about Element
Press 2 to print paricular detail about Element
Press 3 to draw Electron Shell Diagram of given Atomic No.
Enter value1
Enter Atomic No.:5
"Name": Boron
"Appearance": black-brown
"Atomic Mass": 10.81
"Boil": 4200
"Category": metalloid
"Density": 2.08
"Discovered by": Joseph Louis Gay-Lussac
"Melt": 2349
"Molar heat": 11.087
"Named by": null
"Number": 5
"Period": 2
"Group": 13
"Phase": Solid
"spectral_img": null
"Summary": Boron is a metalloid chemical element with Symbol B and atomic Number
5. Produced entirely by cosmic ray spallation and supernovae and not by stellar
nucleosynthesis, it is a low-abundance element in both the Solar system and the
Earth's crust. Boron is concentrated on Earth by the water-solubility of its mor
e common naturally occurring compounds, the borate minerals.
"Symbol": B
"Shells": [2, 3]
"Electronic configuration": 1s2 2s2 2p1
"Electron configuration semantic": [He] 2s2 2p1
"Block": p
```

- Specific detail about element displayed-

```
= RESTART: D:\chempro.py
What you want to print
Press 1 to print All detail about Element
Press 2 to print particular detail about Element
Press 3 to draw Electron Shell Diagram of given Atomic No.
Enter value2
Enter Atomic No.:5
What do you want to print
['Name', 'Appearance', 'Atomic Mass', 'Boil', 'Category', 'Density', 'Discovered
By', 'Melt', 'Molar heat', 'Named_by', 'Number', 'Period', 'Group', 'Phase', 'S
ummary', 'Symbol', 'Electron configuration']
Enter the Key:Boil
Boil:4200
```

- Shell diagram-

Electron shell Diagram is:-



Testing Approach

- **Unit Testing (Data):** Verified atomic numbers 1 (Hydrogen), 50 (Tin) to ensure the Dictionary contains valid keys and values.

```
What you want to print
Press 1 to print All detail about Element
Press 2 to print particular detail about Element
Press 3 to draw Electron Shell Diagram of given Atomic No.
Enter value1
Enter Atomic No.:1
"Name": Hydrogen
"Appearance": colorless gas
"Atomic Mass": 1.008
"Boil": 20.271
"Category": diatomic nonmetal
"Density": 0.08988
"Discovered by": Henry Cavendish
"Melt": 13.99
"Molar heat": 28.836
"Named by": Antoine Lavoisier
"Number": 1
"Period": 1
"Group": 1
"Phase": Gas
"Electronic configuration": 1s1
"Block": s
```

Verification of Atomic Number- 1 (Hydrogen)

```

What you want to print
Press 1 to print All detail about Element
Press 2 to print paricular detail about Element
Press 3 to draw Electron Shell Diagram of given Atomic No.
Enter value1
Enter Atomic No.:50
"Name": Tin
"Appearance": silvery-white (beta,  $\beta$ ) or gray (alpha,  $\alpha$ )
"Atomic Mass": 118.7107
"Boil": 2875
"Category": post-transition metal
"Density": 7.365
"Discovered by": unknown, before 3500 BC
"Melt": 505.08
"Molar heat": 27.112
"Named by": null
"Number": 50
"Period": 5
"Group": 14
"Phase": Solid
"Summary": Tin is a chemical element with the Symbol Sn (for Latin:stannum) and atomic Number 50. It is a main Group metal in Group 14 of the Periodic table. Tin shows a chemical similarity to both neighboring Group-14 elements, germanium and lead, and has two possible oxidation states, +2 and the slightly more stable +4.
"Symbol": Sn
"Shells": [2, 8, 18, 18, 4]
"Electronic configuration": 1s2 2s2 2p6 3s2 3p6 4s2 3d10 4p6 5s2 4d10 5p2
"Electron configuration semantic": [Kr] 4d10 5s2 5p2
"Block": p

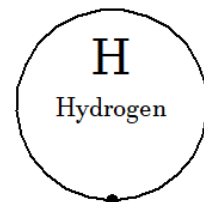
```

Verification of Atomic Number- 50 (Tin)

- **Functional Testing (Drawing):**

- Tested Atomic No. 1: Verified 1 circle, 1 dot.

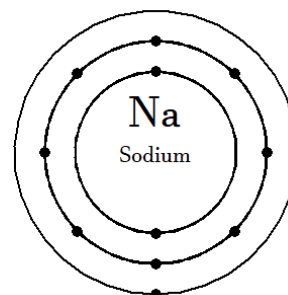
Electron shell Diagram is:-



Verification of Atomic No.-1

- Tested Atomic No. 11: Verified 3 circles (2, 8, 1 config).

Electron shell Diagram is:-



Verification of Atomic No.- 11

Challenges Faced

1. **Screen Resolution scaling:** Using turtle with fixed coordinates (goto(-705, 100)) caused issues on smaller screens. This was mitigated by adding pyautogui to maximize the window.
2. **Electron Spacing:** Mathematically calculating the angle to place electrons evenly around circular orbit was tricky to implement inside the drawing loop.
3. **Data Entry:** Manually entering data for 118 elements was time-consuming and was prone to multiple typo errors.

Learnings & Key Takeaways-

- **Data Structures:** Gained deep understanding of nested dictionaries in Python.
- **Coordinate Geometry:** Learned how to manipulate turtle and movement to draw complex circular patterns.
- **Library Integration:** Learned how to mix console-based logic with GUI-based outputs (turtle + tkinter).
- **Function use:** Learned how to use function in python.

Future Enhancements-

1. **GUI Interface:** Replace the console input with a tkinter or PyQt form for a more modern look.
2. **3D Visualization:** Using OpenGL or Python to show 3D orbitals instead of 2D Bohr models.
3. **API Integration:** Fetch element data from a live chemistry API instead of hardcoding it, ensuring data is always up to date.
4. **Exception Handling:** Improve input validation to handle non-integer inputs gracefully.

References-

1. **Python Documentation:** <https://docs.python.org/3/>
2. **Turtle Graphics Docs:** <https://docs.python.org/3/library/turtle.html>
3. **Wikipedia:** Source of element data (Mass, Density, Discovery info) as noted in source code comments.