Product Lifecycles with Python	Grade 9 - Chemistry

Lesson Plan		
	Coding Tool	Python
	Cross-curricular	Coding

Big Ideas

C1. Relating Science to Our Changing World
assess social, environmental, and economic impacts of the use of elements, compounds, and associated technologies

A1. STEM Investigation Skills
apply scientific processes and an engineering design process in their investigations to develop a conceptual understanding of the science they are learning, and apply coding skills to model scientific concepts and relationships

Specific Expectations

C1.1 assess social, environmental, and economic impacts of processes associated with the life cycle of consumer products, considering the elements and compounds used to make them, and suggest ways to enhance positive impacts and/or minimize negative impacts

C1.2 analyse impacts of using emerging chemical technologies in various fields, including in the skilled trades, and assess factors that influence the development of these technologies

Description

This program helps students understand how products are made, used, and recycled by combining science and coding. Students use Python to model product lifecycles and explore new chemical technologies, all while building basic coding skills in a hands-on, practical way.

Materials

- Computer
- Graphing Tool

Computational Thinking Skills

- Variables
- Loops
- Data visualization

• Python IDE (Visual Studio)	• Simple simulations

Introduction

Everyday products—like water bottles, phones, and clothing—are the result of chemical choices about materials, energy, and how something can be reused or recycled. In this lesson, students explore the "lifecycle" of a product from raw material extraction to disposal, and consider the social, environmental, and economic impacts that show up at each stage.

Students will then use Python in Visual Studio Code to model and visualize lifecycle data, building foundational coding skills (variables, loops, and simple data analysis) while making evidence-based recommendations for reducing impact.

Action

Minds on activity:

Have students select a common, everyday product, such as a plastic water bottle, a smartphone, or a cotton T-shirt. Then collaboratively investigate its complete lifecycle.

Your class begins by identifying the materials used to make the product, considering their natural sources and how they are extracted or harvested. Students research where the raw materials originate, the manufacturing processes involved, and the energy required for production.

They should also discuss how the product is transported, sold, and ultimately used by consumers, noting any factors that affect its lifespan.

Finally, have the students explore what happens to the product after its useful life, examining options for reuse, recycling, or disposal, and considering the environmental impacts at each stage.

Python Modeling:

Step 1: Install Python

Python is the programming language used for this lesson. Here's how to install it:

- Visit the official Python website: https://www.python.org/
- Click Download Python (choose the latest version for your operating system: Windows, macOS, or Linux).
- Run the installer. Important: On Windows, check the box "Add Python to PATH" before clicking "Install Now".
- Follow the prompts to complete installation.
- To confirm installation, open your computer's Command Prompt (Windows) or Terminal (macOS/Linux) and type python --version. You should see a version number (e.g., Python 3.10.0).

Step 2: Install Visual Studio Code (VS Code)

VS Code is a free, user-friendly editor for writing and running Python code.

- Go to the VS Code website: https://code.visualstudio.com/
- Download and install VS Code for your operating system.
- Once installed, open VS Code.

Step 3: Set Up a New Python File in VS Code

Open VS Code.

- Click File > New File (or File > Open Folder to create a folder for your project).
- Save your new file as product_lifecycle.py (or any name ending with .py).
- Make sure you have the Python extension installed in VS Code. If a prompt appears, click "Install" or search "Python" in the Extensions sidebar and install the Microsoft Python extension.

Step 4: Install the matplotlib Library

The code uses matplotlib to create charts. Here's how to install it:

- In VS Code, click View > Terminal to open a terminal window at the bottom of the screen.
- Type the following command and press Enter:
- pip install matplotlib
- Wait for the installation to finish. You should see messages indicating success.

Step 5: Copy and Paste the Provided Python Code

Copy the entire code provided in the lesson plan below these instructions.

- Paste it into your .py file in VS Code.
- Save your file (File > Save or press Ctrl+S).

Step 6: Run the Python Code in VS Code

With your .py file open, look for a small “Run” triangle button at the top right (or right-click in the editor and select “Run Python File in Terminal”).
The program will start running in the terminal window below.

Step 7: Interact with the Program and View Results

The program will show a list of materials (e.g., Plastic, Glass, Aluminum) and ask you to choose one, enter “custom”, or type “exit”.

- If you select a material, you’ll see information about it and a chart showing energy use across its lifecycle stages (e.g., Extraction, Manufacturing, Distribution, Usage, Disposal/Recycling).
- If you choose “custom”, you can enter your own material, add a description, and input energy values for each stage.
- After viewing the chart, the program will ask if you want to look up another material or exit.

Step 8: Troubleshooting Common Issues

- Python not found: Make sure you installed Python and added it to PATH during setup.
- Matplotlib not installed: Double-check you ran pip install matplotlib in the VS Code terminal.
- Code errors: Confirm you copied the code exactly. Typos or missing lines may cause issues.
- Program doesn’t run: Make sure you’re running the correct file and have the Python extension installed.
- Chart doesn’t appear: Sometimes charts open in a separate window; check behind other windows or your taskbar.

In the sample code below we have several “preprogrammed” materials, these materials are placeholders to give your students an idea of how to format their findings, once this is coded students should be encouraged to add their own products or materials with their own data to display the amount of MJ or Mega Jules of energy is used in each stage of their materials or products.

Sample Code:

```
# Import the matplotlib library for plotting graphs
import matplotlib.pyplot as plt
```

```
# Preprogrammed materials and their data
materials = {
   # Each material is a dictionary entry with info and energy usage per stage
   "Plastic": {
      "info": "Plastic is made from petroleum. It is lightweight but can take hundreds of years to
decompose.",
      "energy": [10, 25, 5, 2, 8]  # Energy used (in MJ) for each lifecycle stage
   },
   "Glass": {
      "info": "Glass is made from sand. It is recyclable and can be reused many times.",
      "energy": [15, 20, 7, 2, 5]
   },
   "Aluminum": {
      "info": "Aluminum is made from bauxite ore. It is highly recyclable and saves energy when
reused.",
      "energy": [20, 30, 10, 3, 4]
   }
}



# List of product lifecycle stages, used for both chart labels and user input
stages = ["Raw Material Extraction", "Manufacturing", "Distribution", "Usage",
"Disposal/Recycling"]



# Main menu function for user interaction
def main_menu():
   while True:  # Loop until the user chooses to exit
      print("\nAvailable materials:")
      # Print each material name
      for name in materials:
         print(f"- {name}")
      print("- custom (enter your own material)")
      print("- exit (quit the program)")

      # Prompt user to select material, custom, or exit
      choice = input("Type the name of a material from the list above, 'custom' to enter your own,
or 'exit' to quit: ").strip()
```

```
if choice.lower() == "exit":  # Exit condition
    print("Goodbye!")
    break
elif choice in materials:  # If user selects a preprogrammed material
    info = materials[choice]["info"]
    energy = materials[choice]["energy"]
    material = choice
elif choice.lower() == "custom":  # If user wants to enter a custom material
    material = input("Enter the material name: ")
    info = input(f"Enter a brief description for {material}: ")
    energy = []
    print(f"Enter the energy used (in MJ) for each stage of {material}:")
    # Prompt for energy value for each lifecycle stage
    for stage in stages:
        value = float(input(f"  {stage}: "))
        energy.append(value)
else:  # If input is invalid
    print("Invalid choice. Please try again.")
    continue

# Display selected material and its info
print(f"\nMaterial: {material}")
print(f"Info: {info}")

# Plot energy usage for each lifecycle stage using a bar chart
plt.bar(stages, energy, color='skyblue')
plt.xlabel('Lifecycle Stage')
plt.ylabel('Energy Used (MJ)')
plt.title(f'Energy Use Across Product Lifecycle: {material}')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# Ask if the user wants to see another material
again = input("Would you like to look up another material? (yes/no): ").strip().lower()
if again != "yes":  # Exit loop if user says no
    print("Goodbye!")
    break
```

```
# Run the menu function if script is executed directly
if __name__ == '__main__':
    main_menu()
```

Consolidation/Extension

Students can now use this code to add all sorts of materials and calculate their lifecycle emissions.

Assessment

Adding more materials and keeping research notes on materials provide excellent materials for assessment.

Additional Resources

https://www.python.org/
https://calculator.dev/environment/lca-calculator/
https://code.visualstudio.com/