datatypes_fun

May 28, 2024

1 Fun with DataTypes

File as PDF

1.0.1 1. String Manipulation: Physics Puns and Jokes

- **Exercise:** Write a Python program that stores physics jokes or puns in strings and then prints them out to the console. For example, “Why can’t you trust an atom? Because they make up everything!” This introduces into string data types and basic input/output operations.

```
you
```

1.0.2 2. Integers and Floats: Calculating Physics Constants

- **Exercise:** Create a program that calculates and prints the value of various physics constants, such as the speed of light in a vacuum, Planck’s constant, or the gravitational constant. This will help practice using integers and floats, as well as basic arithmetic operations in Python.

Speed of light

- Definition: The speed at which light travels in vacuum, a fundamental physical constant denoted by c.
- Value in Metric **299,792,458 metres per second**
- Upper Speed Limit: c is the maximum speed at which all conventional matter, energy, or any signal carrying information can travel.
- Historical Measurement: First demonstrated to not be instantaneous by Ole Rømer in 1676 using Jupiter’s moon Io.
- Relevance Beyond Light: Albert Einstein showed its significance outside of light in the theory of relativity, including in the equation

  $$E = mc^2$$

  .
- Refractive Index: The speed of light in materials such as glass or air is less than c, affecting its speed based on the medium’s refractive index.

```
Speed of Light in Vacuum: 299792458 m/s
Planck's Constant: 6.62607015e-34 J.s
```

```
Gravitational Constant: 6.6743e-11 N m^2 / kg^2
Boltzmann Constant: 1.380649e-23 J/K
```

1.0.3 3. Lists: Tracking Particles in an Accelerator

- **Exercise:** Create a list of particles (e.g., protons, neutrons, electrons) being accelerated in a hypothetical experiment. They should write functions to add, remove, and modify particles in the list, simulating real-world data manipulation. This introduces lists and list operations.

```
Added 'muon' to the list.
Current list of particles: ['proton', 'neutron', 'electron', 'positron', 'muon']
Removed 'positron' from the list.
Replaced 'muon' with 'tau'.
Current list of particles: ['proton', 'neutron', 'electron', 'tau']
```

1.0.4 4. Tuples: Storing Atomic Data

- **Exercise:** One can use tuples to store atomic data, such as atomic number, atomic mass, and electron configuration for different elements. Write a function to print this data in a formatted way. This exercise teaches the immutability of tuples and how they can be used to store related data.

```
 [(1, 1.008, '1s1'),
  (2, 4.002602, '1s2'),
  (6, 12.011, '[He] 2s2 2p2'),
  (8, 15.999, '[He] 2s2 2p4')]
```

1.0.5 5. Dictionaries: Cataloging the Periodic Table

- **Exercise:** Create a dictionary where each key-value pair consists of an element (key) and its properties (value) such as atomic number, atomic mass, and state at room temperature. This exercise introduces dictionaries and how to access and modify their data.

```
Properties of Carbon (C):
Atomic Number: 6
Atomic Mass: 12.011
State at Room Temperature: solid

Updated properties of Carbon (C):
State at Room Temperature: graphite (solid)

Added properties of Neon (Ne):
Atomic Number: 10
Atomic Mass: 20.1797
State at Room Temperature: gas
```

1.0.6 6. Boolean Logic: Evaluating Collision Outcomes

- **Exercise:** Write a program that uses Boolean logic to determine the outcome of particle collisions based on their properties (e.g., mass, velocity). For example, whether the particles will bounce off each other, merge, or disintegrate. This is about Boolean data types and conditional statements.

```
Outcome of the collision:
Bounce: False
Merge: True
Disintegrate: False
```

1.0.7 7. Sets: Unique Quantum States

- **Exercise:** Create a set of quantum states that a particle can occupy, then write functions to add and remove states, ensuring no duplicates are allowed. This can introduce the concept of sets and their properties, such as uniqueness and set operations.

```
Initial quantum states: {'state1', 'state2', 'state3'}
Quantum states after adding new states: {'state1', 'state4', 'state2', 'state3'}
Quantum states after removal: {'state4', 'state2', 'state3'}
```

1.0.8 8. Fun with Complex Numbers: Quantum Mechanics Basics

- **Exercise:** Since complex numbers are used in quantum mechanics, perform operations with complex numbers in Python (e.g., adding wave functions). This will help to get comfortable with complex data types and their applications in physics.

Indistinguishable particles

In quantum mechanics, indistinguishable particles are particles that cannot be distinguished from one another, even in principle.

- Categories of particles: Bosons and fermions
- Principle: Cannot be distinguished from each other even in principle
- Examples of bosons: Photons, gluons, helium-4 nuclei
- Examples of fermions: Electrons, neutrinos, protons

Indistinguishability has important consequences in quantum mechanics in the same way as it is of importance for probability theory.

```
Wave Function 1: (3+4j)
Wave Function 2: (1-2j)
Sum of Wave Functions: (4+2j)
Difference of Wave Functions: (2+6j)
Product of Wave Functions: (11-2j)
Quotient of Wave Functions: (-1+2j)
Conjugate of Wave Function 1: (3-4j)
Resultant Wave Function (psi1 + psi2): (6-2j)
```