**Main Class (App)**

* **Purpose**: It’s used to collect data for a computer object and then display the computer’s information.
* **Process**:
  + The program asks the user to input information about different computer components (CPU, RAM, HDD, Motherboard).
  + For each component, it has a method (getCpuData, getRamData, etc.) that prompts the user for various attributes.
  + It ensures that the input is valid (e.g., positive numbers for sizes and years, specific options for types like "DDR" for RAM).
  + After collecting the data, it creates a Computer object using all the collected details.
  + The final result is displayed using the toString() method of the Computer class, which returns a nicely formatted description of the computer.

**Classes for Computer Components**

1. **CPU Class**:
   * **Attributes**: clock\_frequency, instruction\_set, type (32 or 64-bit), number\_of\_registers, and manufacturer.
   * **Methods**:
     + A constructor is provided to initialize all attributes. There’s also a default constructor with default values (0, "NULL", etc.).
     + The toString() method formats the CPU data for easy display.
   * **Purpose**: It represents the CPU’s basic characteristics.
2. **RAM Class**:
   * **Attributes**: size (GB), type (DDR, DDR2, DDR3), manufacturer.
   * **Methods**:
     + Constructor initializes these values. There’s also a default constructor with placeholder values.
     + The toString() method formats the RAM data for easy display.
   * **Purpose**: It represents the RAM details for the computer.
3. **HDD Class**:
   * **Attributes**: size (GB), rotations (RPM), manufacturer.
   * **Methods**:
     + Constructor initializes values and a default constructor is also provided.
     + The toString() method formats the HDD details.
   * **Purpose**: It represents the Hard Drive details for the computer.
4. **Motherboard Class**:
   * **Attributes**: chipset\_version, usb\_ports, ram\_capacity (GB), manufacturer.
   * **Methods**:
     + Constructor initializes the attributes and a default constructor provides default values.
     + The toString() method formats the motherboard details for display.
   * **Purpose**: It represents the Motherboard's essential information.

**Computer Class**

* **Attributes**:
  + It stores the CPU, RAM, HDD, and Motherboard objects, as well as the assembly\_year and price.
* **Constructor**:
  + Initializes all attributes, including the components of the computer.
* **Methods**:
  + The toString() method is overridden to return a detailed and formatted summary of the computer, including all its parts and details.
* **Purpose**: It aggregates all components into a single computer object.

**User Interaction**

* **Input Validation**:
  + For each component, there are validation checks. If the user enters an invalid value (e.g., a non-positive number for sizes or price), the program prompts them to re-enter a valid value.
  + This is done using while (true) loops to keep asking until the input is valid.
* **Output**:
  + After gathering all the data, the program creates a Computer object and displays the details in a formatted string using System.out.println(test).

**Thoughts While Coding:**

* **Handling Input Validity**: Ensuring the user inputs valid data is important to avoid errors during object creation. This leads to the use of while loops and validation checks.
* **Structured Data**: Each computer component is encapsulated in its class (like CPU, RAM, HDD, etc.), keeping the code clean and organized.
* **Readability**: By overriding the toString() method, it becomes easy to display the objects in a readable format without needing to manually format each component when displaying the Computer object.