Project Planning and Design:

- Analyze the problem and break it down into smaller, manageable parts.
- Define the classes and their relationships (Inheritance, Composition, etc.).
- Plan out the data structures and algorithms needed to implement the simulation.

Development Steps:

1. **Create the Grid:**
   - Develop a 20x20 grid that can hold critters (ants and doodlebugs).
   - Implement checks to ensure critters don't move off the grid.
2. **Define the Organism Class:**
   - Create a base class `Organism` with common data and functions like `move()`.
   - The `move()` function should be virtual and overridden by derived classes.
3. **Create Ant and Doodlebug Classes:**
   - Derive `Ant` and `Doodlebug` classes from the `Organism` class.
   - Implement the specific behaviors for each critter as described in your assignment.
4. **Initialize the World:**
   - Place 5 doodlebugs and 100 ants randomly within the grid.
5. **Implement Simulation Logic:**
   - Code the logic for each time step, including moving, breeding, and starving.
   - Ensure that doodlebugs move before ants in each turn.
6. **Drawing the World:**
   - Represent ants as “o”, doodlebugs as “X”, and empty spaces as "-".
   - Draw the ASCII grid in the console after each time step.
7. **User Interaction:**
   - After each time step, prompt the user to press Enter to proceed to the next time step.
8. **Testing and Debugging:**
   - Test the program to make sure all rules are correctly implemented.
   - Debug any issues that arise during testing.
9. **Optimization:**
   - Review the code for efficiency and performance optimization.
   - Clean up and refactor the code if necessary.
10. **Documentation:**
    - Comment your code to explain the logic and flow.

- Provide instructions on how to compile and run the simulation.

Absolute Requirements:

- The world must be a 20x20 grid.
- Only one critter per cell is allowed.
- Implement the exact behaviors for moving, breeding, and starving for both ants and doodlebugs.
- Doodlebugs move before ants in each turn.
- The grid should be drawn using ASCII characters with the specified representations for ants, doodlebugs, and empty spaces.
- The user should prompt the simulation to move to the next time step.
- Submit a single `.cpp` file that includes all classes and is executable.
- No separate `.h` files are allowed; everything must be in one `.cpp` file.