In the context of the Harris Hawk Optimization (HHO) class and the **exploitPhase** function, the exploitation phase aims to refine the solutions found during the exploration phase by intensifying the search around the best solutions. Here's a high-level outline of how you might implement the **exploitPhase** function based on the HHO algorithm:

1. Initialize the best solution found so far (**bestPlacement**) and the worst solution (**worstPlacement**).
2. Calculate the best fitness value (**bestFit**) and the worst fitness value (**worstFit**).
3. For each iteration, perform the following steps:  
   a. Select a random hawk (**y\_rand**) from the population (**pop**).  
   b. Calculate the distance between the random hawk and the best solution (**dist\_best**).  
   c. If **dist\_best** is less than a predefined threshold (e.g., 10% of the search space size), perform a local search around the best solution.  
   d. Update the best and worst solutions if the new solution has a better fitness value.  
   e. Update the number of function evaluations (**numFuncEval\_ga**).  
   f. Update the cross-over and mutation counts (**crossOverCount** and **mutationCount**).

This is a high-level outline, and the actual implementation would require more details and specific functions from the HHO class. For example, you might need to implement functions for local search, calculating distances, and updating the population.Please note that the provided outline is not a complete implementation of the HHO algorithm, but rather a high-level guide to help you understand how to implement the **exploitPhase** function. For a more detailed implementation, you might want to refer to the HHO algorithm's original paper or other resources that provide a more comprehensive explanation.