Robot

// Robot follows the open close principle because the robot class can easily be changed and

// different functions can be added in the future and the three principles main will not be

// changed, Robot also follows the the single responsibility because Robots only have the

// responsibility of working therefore following the single responsibility principle.

// The Interface Segregation Principle does not apply to the Robot class because it is not an

// interface

Super Worker

// Super Worker follows the single responsibility principle because it implements the Iworker

// interface, it only has the methods that a super worker object needs. Therefore it follows the

// principle. Super Worker also follows the open close principle because super worker has no

// effect on the existing source code, but still can be modified and methods cans till be added in

// the future if the super worker needs to do more. The Interface segregation Principle does not

// apply to the Super Worker because it is not an interface.

IWorker

// The IWorker class follows the open close principle because the IWorker class can easily be

// changed and have new functions added in the future, thus it is open to expansion, but doesn’t

// require having its previous functions be modified to do so, thus it is closed to modification. It

// also follows the interface segregation principle because rather than being an interface packed

// with a lot of functions and purposes, making it a more general interface, it follows a specific

// purpose, defining the actions a worker can take, and is also separated from the other

// interfaces in the package. The single responsibility principle is also followed as the class

// provides functions that only serve to help the worker class.

Manager

// The Manager class follows the open close principle because the Manager class can easily be

// changed and have new functions added in the future, thus it is open to expansion, but doesn’t

// require having its previous functions be modified to do so, thus it is closed to modification. The

// single responsibility principle is also followed as the class provides functions that only serve to

// manage workers, thus geared for managers only.

ThreeSolidMain

// The ThreeSolidMain Class follows the Single Responsibility Principle of running the program

// when it is called by the Operating System. This is due to the fact that the main class serves

// only one type of user -- the person running the program. The I/O is geared to serve the user.

// Similarly, the ThreeSolidMain Class also follows the Open Close Principle because the main

// class calls for the Manager class instead of implementing setWorker() and Manage() methods.

Worker

//Worker implements IWorker and follows the open close principle. Worker also follows the

//single responsibility principle because it the only reason it has to change is with functionality of

//worker.

IFeedable

// IFeedable follows the Interface segregation principle because it is only the interface.

// It also follows the single responsibility principle by only having the eat method

IWorkable

// IWorkable follows the Interface segregation principle because it is only the interface,

// It also follows the single responsibility principle by only having the work method