Project plan

Requirements

Need to write a **server.**

The server solves problems.

The server side:

The server will run with this command:

```
./ex4.out <port> [server type]
```

The server will listen on *port*, and will operate *parallel* or *serial* (as specified in server type)

The client side:

The client will send a message in this format:

```
solve <problem> [algorithm]
<two line breaks>
```

If the server received the message, it will return a reply.

Next, the client will send the **graph** in the specified format

The server tries solving the problem and returns a suitable message.

Design

Abstract class **server:**

open(port, ClientHandler) – open the server on this port and listen to clients

close() – close this server.

Class **SerialServer** implements server.

Class **ParallelServer** implements server.

Abstract class **ClientHandler –** handles the client input and returns the necessary output:

handleClient(inputStream, outputStream)

Class **AlgorithmClientHandler** implements ClientHandler:

solver: Solver

cacheManager: CacheManager

Class **DFSSolver, BFSSolver, BestFSSolver, AStarSolver** implement **solver** interface, and they will choose The parameters <ProblemType, SolutionType> for the Solve method .

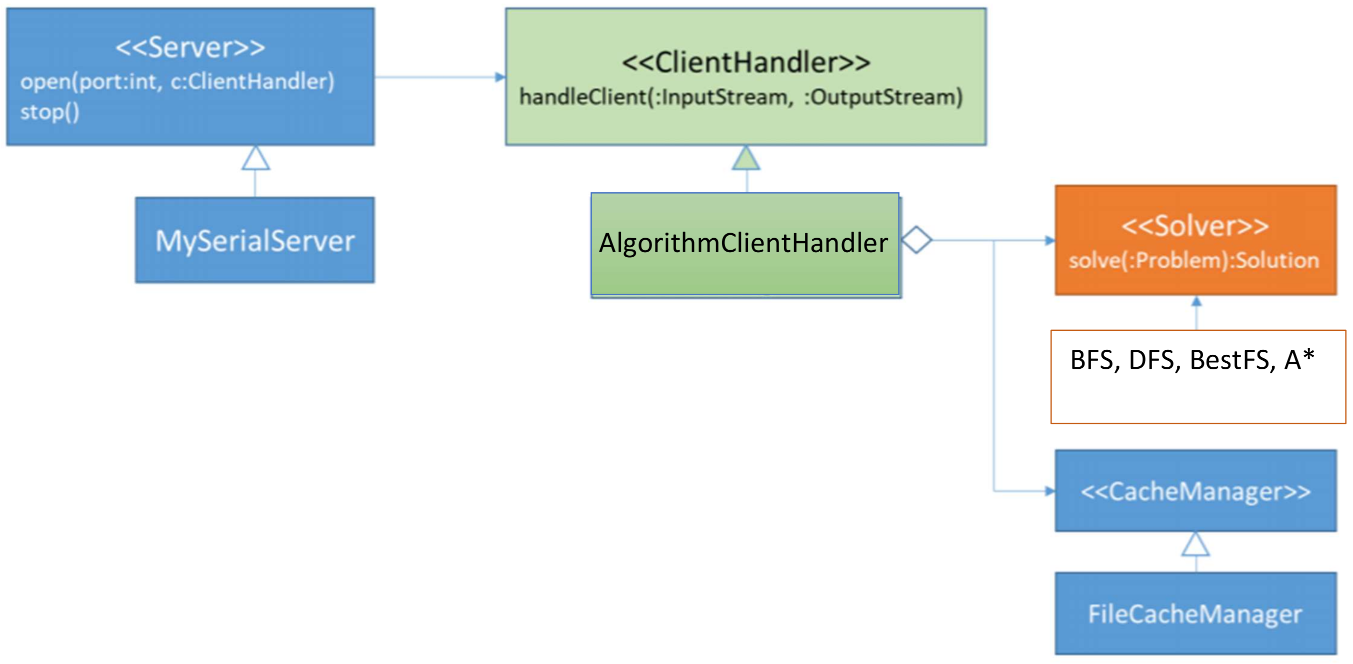
Abstract class **Solver:**

Solve<ProblemType, SolutionType>()

this class will handle the algorithm client according to the solver (DFS, BFS ,BestFS ,A*,..)

the class will use a cache manager to save uneccesary calculations.

The final hierarchy:



Coding

Bottom – up approach:

1. solver
    - abstract class
    - BFS, DFS,.. solver classes.

----

2. Client Handler
    - abstract class
    - algorithm handler class

----

3. Server
    - abstract class
    - SerialServer class
    - ParallelServer class