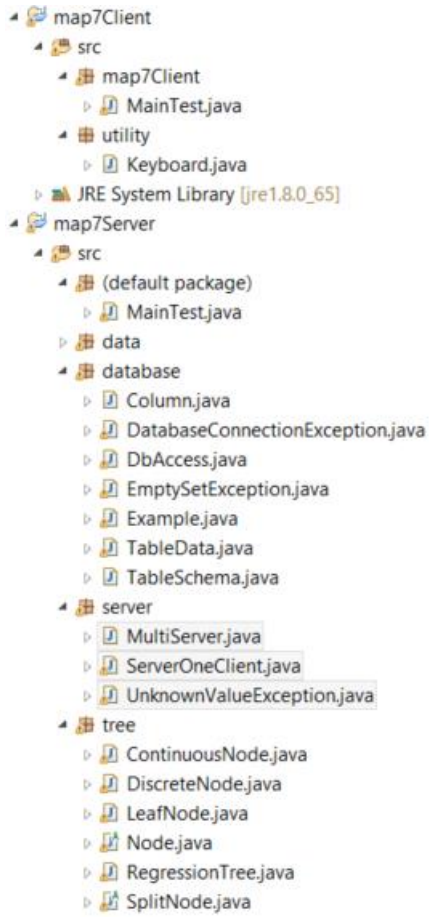
Esercitazione 7– Client–Server (Socket)

Creare due progetti Eclipse distinti, mapClient e mapServer. Distribuire le classi finora definite tra i due progetti.



CLIENT

Il sistema client deve collegarsi al server tramite l'indirizzo e la porta su cui il server è in ascolto.

Indirizzo del server e porta sono acquisiti come parametri tramite args. Una volta instaurata la connessione l'utente può scegliere se avviare un nuovo processo di clustering o recuperare cluster precedentemente serializzati in un qualche file.

Includere la classe MainTest (fornita dal docente) che stabilisce la connessione al Server e, una volta avvenuta la connessione, invia e riceve messaggi, dipendentemente dalla scelta effettuata dall'utente.

(FORNITA DAL DOCENTE)

SERVER

Il server colleziona le classi per la esecuzione remota del processo di data mining

Definire il package Server che contiene le classi Server, ServerOneClient e UnknownValueException (prima contenuta nel package tree).

- Definire la classe MultiServer

Attributi

private int PORT = 8080;

Metodi

public MultiServer(int port): Costruttore di classe. Inizializza la porta ed invoca run()

private void run**()** Istanzia un oggetto istanza della classe ServerSocket che pone in attesa di crichiesta di connessioni da parte del client. Ad ogni nuova richiesta connessione si istanzia ServerOneClient.

- Definire la classe ServerOneClient estendendo la classe Thread.

  Attributi

```
private Socket socket;
private ObjectInputStream in;
private ObjectOutputStream out;
```

  Metodi

  public ServeOneClient**(Socket s) throws IOException**: Costruttore di classe. Inizializza gli attributi socket, in e out. Avvia il thread.

  public void run**()** Riscrive il metodo run della superclasse Thread al fine di gestire le richieste del client in modo da rispondere alle richieste del client.