SocioNet

Design:

- Data interface
  - To provide inbuilt data
  - The purpose of the same is so that the main method class is unaware of the data source
  - As it only contains data, it is type of an abstract
- MiniNet Class()
  - This is defined only to run our network and contains no instance variables
  - Executes only the main method
- Abstract Driver Class()
  - This is the class where all the action happens
  - Here we have used HashMap<Integer, Profile>
    - To contain the information of profiles
    - Integer is being used to define the unique ID for each profile
    - Profile is an abstract class created for information
  - All the methods such as addProfile(), updateProfile() are defined in this class
- Abstract Profile Class()
  - This class is created to define instance variables such as name, age, status, photo
  - This class also contains the instance variable ArrayList<String> friendlist to contain the names of friends
  - Also, while entering the status, we may pass "_" value but never a null value
  - This is the super class for the following two classes:
    - Adult
    - Child
- Adult Class()
  - Based on the age > 16, the profiles will be stored with reference to this class
  - This class also contains an instance variable of ArrayList<String> childlist to contain the name of the children who are on SocioNet
- Child Class()
  - Based on age>2 && age <16, the profiles will be stored with reference to this class
  - This class contains an instance variable String[] Dependent:
    - Size: 2
    - Contains the names of the parents if they are on SocioNet
    - This is the mandatory requirement for children if they want to create a profile

Assumptions:

While designing the application, we have taken a few assumptions:

- No child will have a single parent
- While updating the dependents/parents, child has to enter the names of both dependents again
- A user can enter a string for variables such as name, status in capital or small letter, but we will store all the information in small letters
- All the people in our database will have unique names as our search will return the first name it encounters in the database while displaying profile
- The unique ID will be used to access any profile and we have taken 9999 as the maximum number of users in our application for now
- We have started the data with profile having unique ID, 1001 to 1008
- Although we have given an instance variable for photo, but for simplicity, we are not using the same on our network and thus not asking the user to add the same
- Status cannot be null, but it can be "_"

CLASS DIAGRAM