Friendly Search:

People typically have friends who are similar to themselves in one or more ways such as age, school, workplace, hobbies, or interests. Because a person has something in common with his friends, he is more likely to be interested in videos that his friends like, rather than videos that strangers like. Similarly, a person is more likely to have something in common with a friend of a friend, rather than a stranger. This increased probably of having something in common is likely to continue through several degrees of friends, with a decreasing likelihood of commonalities at each step. To take advantage of this and provide better search results, Friendly Search searches YouTube for videos related to a given search phrase, and uses a person's social network to order the search results.

The Friendly Search algorithm takes as input:
1. A query string to search for
2. A social network stored in a dictionary and consisting of user-names of people and user-names of people they are friends with
3. A dictionary of what YouTube videos each user in the social network has "Liked".
4. The name of the user who has requested the search

The Friendly Search algorithm considers 5 degrees of friendship in its search rankings. In other words, it starts with the user requesting the search, looks at that user's immediate friends, looks at those users' friends, and continues until it has looked at the 5th degree of friends (i.e. a friend of a friend of a friend of a friend of a friend). Because the user is more likely to have common interests with a lower-degree friend than a higher-degree friend, the algorithm will rank a video "Liked" by a lower-degree friend higher than a video "Liked" by a higher-degree friend.

The Friendly Search algorithm outputs a list of YouTube videos, with the following information about each one:
1. Title
2. Url
3. YouTube Rating (based on how YouTube users have rated the video)
4. Friendly Rating (based on the user's friends and "Likes" within the social network)
5. A List of the User's Immediate Friends Who Have "Liked" the Video

The videos are ordered first by the friendly rating. For those videos with a friendly rating of 0, the videos are then ordered by the YouTube rating.

The social network dictionary and the likes dictionary are currently hard-coded to demonstrate how the algorithm works. I would like to get this on the web eventually, and use the Facebook API to get the user's social network. The YouTube video search results would be embedded on the website & there would be a button next to each video for the user to "Like" the video. Those "Likes" would then be stored in a database. Next to each video in the search results there would be profile photos of the user's immediate friends who have "Liked" the video.

The friendly search code is stored in FriendlySearch.py. The atom and gdata folders contain files for the Google Python Data Client Library, which is FriendlySearch.py uses to access the YouTube API.