Fundamental Algorithms, Spring 1998

Assignment 9. **Greedy Algorithms.**

Given April 1, due Monday, April 13.

1. (from CLR) The greedy strategy for making change is to build up a collection of coins by repeatedly adding the most valuable coin that keeps the total from exceeding the desired total.

a. Give an example of a set of coin values (the v_k from HW 8) for which this strategy can fail to produce the fewest coins adding up to a given total.

b. Someone suggested that the greedy strategy always works (gives the minimal number of coins) if each coin is at least twice as valuable as the most valuable coin of lesser value ($v_k \geq 2 \cdot v_{k-1}$). Show that this is also not so.

c. What about the case $v_1 = 1$, $v_2 = 5$, $v_3 = 10$, $v_4 = 25$?

2. Each person has a taste in music. In our computer, this is represented by letting the variable `person.taste` have the values `ROCK` or `JAZZ`. A person is an object of data type `individual`. There are `N` people, in an array `individual people[N];`. Each person has a number of friends, stored in `person.Nfriends`, and a list of friends, stored in the array `person.friends`. If a person has a friend who has a different taste in music, that creates `discord`. The total discord is computed as follows (please excuse the C bugs, I hope the meaning is clear.):

```
\tt
int discord = 0; int M;
individual person, friend;
for ( int i = 0; i < N; i++) {
   person  = people[i];
   M        = person.Nfriends;
   my_taste = person.taste
   for ( int j = 0; j < M; j++) {
      friend      = person.friends[j];
      their_taste = friend.taste;
      if ( my_taste != their_taste ) discord++;
   }
}
```

We have the option to change the taste of `k` people and want to do so to lower the discord as much as possible. Design a greedy strategy to do this. Your strategy should change the taste of one person at a time, at each step switching the person who, at that stage, is creating the most discord. Does this strategy give the optimal switches?