CS6600 Homework 2

Matthew Whitesides
mbwxd4@umsystem.edu

I. Chapter 2 Problems

1) a)

	alicerc	*bobrc*	*cyndyrc*
Alice	own	read	none
Bob	read	own	none
Cyndy	read	read, write	own, read, write

b)

	alicerc	*bobrc*	*cyndyrc*
Alice	own	read	read
Bob	none	own	none
Cyndy	read	read, write	own, read, write

4) a) Remove all q from o.

```
command delete_all_rights(p, q, o)
      delete read from A[q, o];
      delete write from A[q, o];
      delete execute from A[q, o];
      delete append from A[q, o];
      delete list from A[q, o];
      delete modify from A[q, o];
      delete own from A[q, o];
end
```

b) Remove all q from o if p can modify o.

```
 command delete_all_rights(p, q, o)
       if modify in A[p, o]
       then
         delete read from A[q, o];
         delete write from A[q, o];
         delete execute from A[q, o];
         delete append from A[q, o];
         delete list from A[q, o];
         delete modify from A[q, o];
         delete own from A[q, o];
 end
```

c) Remove all q from o if p can modify o and q does not have own rights.

```
 command delete_all_rights(p, q, o)
       if modify in A[p, o] and own not in A[q, o]
       then
         delete read from A[q, o];
         delete write from A[q, o];
         delete execute from A[q, o];
         delete append from A[q, o];
         delete list from A[q, o];
         delete modify from A[q, o];
         delete own from A[q, o];
 end
```

5) a) Copy all rights from p to q.

```
command copy_all_rights(p, q, o)
 if read in A[p, o]
 then
  enter read into A[q, o];
 if write in A[p, o]
```

```
  then
     enter write into A[q, o];
  if execute in A[p, o]
  then
     enter execute into A[q, o];
  if append in A[p, o]
  then
     enter append into A[q, o];
  if list in A[p, o]
  then
     enter list into A[q, o];
  if modify in A[p, o]
  then
     enter modify into A[q, o];
  if own in A[p, o]
  then
     enter list into A[q, o];
end
```

b) Copy all rights from p to q if the right has the copy flag.

```
command copy_all_rights(p, q, o)
  if read in A[p, o] and copy in A[p, o]
  then
    enter read into A[q, o];
  if write in A[p, o] and copy in A[p, o]
  then
     enter write into A[q, o];
  if execute in A[p, o] and copy in A[p, o]
  then
     enter execute into A[q, o];
  if append in A[p, o] and copy in A[p, o]
  then
     enter append into A[q, o];
  if list in A[p, o] and copy in A[p, o]
  then
     enter list into A[q, o];
  if modify in A[p, o] and copy in A[p, o]
  then
     enter modify into A[q, o];
  if own in A[p, o] and copy in A[p, o]
  then
     enter list into A[q, o];
end
```

c) If you copied the copy flag along with the right you would essentially be giving the copied user the ability to copy those rights over to any other user of their choosing.

6) a) Copy r rights from Alice (p) to book to Bob (q).

```
   command copy_right_to_book(r, p, q)
     if c in A[p, book] and r in A[p, book]
     then
       enter r into A[q, book];
   end
```

b) Copy r rights from Alice (p) to book to Bob (q).

```
   command copy_right_to_book(r, p, q)
     if rc in A[p, book]
     then
       enter r into A[q, book];
   end
```

c) If the copy flag is not copied over to Bob, Bob would not have the ability to copy right r over to anyone else. Which I would think would be correct that you would not want to give bob rc by default that might be a separate command.

10) a) If you did not apply the principle of attenuation at all in a system, any subject would be able to grant rights or

increase their own rights by granting them to themselves. Essentially all the rights in the system would be available to any subject, this may only be limited by what rights the subject is aware they could have, so if they could cooperate with other subjects they would potentially be able to have the rights of all unique rights over the set of all subjects. Also if there is a copy ability any subject with read permission could copy a file and grant anyone else full permissions as they would be the owner of the copy.

b) If attenuation was applied to access rights but not own and grant_rights, this would not help the situation as if the subjects with the grant or own permissions were in on it they have the ability to grant any permission to any other subject regardless if they have the right themselves. If the subjects with the grant permission did not have the ability to grant permissions that they themselves do not have like read then that would limit the maximal set to the rights of any subject union with the subjects with grant permission.

c) The following ACM shows two subjects with subject1 being an ancestor of subject2, subject1 has right r which is inherited by subject2. This would lead to the maximal set being the union of any given subject and all its ancestors.

	subject1	*subject2*	*object*
subject1	r	none	r
subject2	r	r	r

ACKNOWLEDGMENT

The author would like to thank Professor Bruce McMillin with the Department of Computer Science, Missouri University of Science and Technology.

Matthew Whitesides Master's Student at Missouri University of Science and Technology.