# BIO508: Lab Session 8

## Announcements

- Homework 7 is due Monday at 11:55pm
- Homework 6 grades will be posted on Friday

## Going over Problem Set 5

### File Grading

The program which I use to identify differences between your files and the key's is called `diff`. It detects *any* differences between files, so your file must be precisely the same as the key's for full credit (no extra lines, etc.).

Don't put asterisks around your file names. For example if the questions asks you to make a function and save it as `*some_function.py*` you should turn in `some_function.py`.

### Questions?

## Going over Problem Set 6

### Useful Python Functions

`set()`

`set()` defines a list-like object which only contains *unique* elements. Note that unlike lists, `set`s are unordered, so you can't index them

```
set([1,1,2,3,2,4,1,5]) ⇒ (1,2,3,4,5)

set([1,2,3]) ⇒ (1,2,3)

set([1,1,3])[0] ⇒ ERROR!!
```

`set.add(elt)`

`set.add(elt)` adds `elt` to `set`. In sets, adding an element means "appending" it only if the element is unique.

```
my_set = set([1,1,4,3,2])

my_set.add(3)

print my_set ⇒ set([1,2,3,4])

my_set.add(7)

print my_set ⇒ set([1,2,3,4,5])
```

`hash.setdefault(key[, default])`

`hash.setdefault(key[, default])` does two things. First, it checks whether `key` is in `hash`. If not, it then adds `key` as a key to `hash` with the value `default`. Second, it returns the value of `hash[key]`. Note that this means that if `key` is not in `hash`, `hash.setdefault(key[, default])` returns `default`.

```
myDict = {'Bug1': 'streptococcus'}

myDict.setdefault('Bug1',"") ⇒ 'streptococcus'

print myDict ⇒ {'Bug1': 'streptococcus'}

myDict.setdefault('Bug2',"") ⇒ ""

print myDict ⇒ {'Bug2': '', 'Bug1':'streptococcus'}
```

## regroup_sequences.py

This problem seemed to confuse several people, so I wanted to go over it all together. Now that we know what the functions do, let's make our own key for `regroup_sequences.py`

```
#!/usr/bin/env python

------ ---
------ ---

if ___( ________ ) != _:
        _____ _________( "Usage: regroup_seqs.py <data.groups> < <rdp.txt>" )
strGroups = ___________

hashGroups = {}
for ________ in csv.______( open( strGroups ), _____________ ):
        __________[astrLine[0]] = astrLine[1]

setstrSamples = set()
hashhashCounts = {}
for ________ in ____reader( _________, _____________ ):
        strSample, strBug = hashGroups[astrLine[0]], ________[1]
        setstrSamples.add( _________ )
        hashBug = hashhashCounts.setdefault( strBug, {} )
        hashBug[strSample] = _ + hashBug.get( strSample, _ )

print( "\t".join( ["Bug \ Sample"] + list(setstrSamples) ) )
for strBug, hashBug in hashhashCounts._____( ):
        astrLine = [strBug]
        for strSample in _____________:
                astrLine.______( ___(hashBug.get( strSample, 0 )) )
        print( "\t".join( astrLine ) )
```