Software needed:
fastqc
stacks

Note about stacks: If you download the most recent version, you will need a newish g++ compiler. If you are having trouble compiling stacks this might be the problem. I had to compile a local version of g++, which means I need to specify the path using "source" each time I use stacks

1. Download files from UC Davis

```
wget -r -nH -np
"http://slims.bioinformatics.ucdavis.edu/Data/2bf8xyfqt8/Unaligned/Project_TSKR_L3_YWAR_Plate2/"
```

This will make a weird folder structure, because all parent directories starting with Data get downloaded. You will have to move the folder to avoid this

```
mv Data/2bf8xyfqt8/Unaligned/Project_TSKR_L3_YWAR_Plate2/ Plate2/
```

2. Run fastqc. Do this step for each fastq file. You will want to download the output to your computer so you can look at the report (html format).

```
zcat Plate1_S2_L002_R1_001.fastq.gz | fastqc stdin
```

3. Run flip script to make sure barcode is always in read 1 for bestRad libraries. This script also trims two leading base pairs "GG".

If your files are zipped (.gz) you will need to first unzip using the command:

```
gunzip WIFL-Plate2_S1_L001_R2_001.fastq.gz
gunzip WIFL-Plate2_S1_L001_R1_001.fastq.gz
```

Then we use the flipscript on the unzipped file:

```
perl flip_trim_werrors.pl barcodes.txt WIFL-
Plate2_S1_L001_R1_001.fastq WIFL-Plate2_S1_L001_R2_001.fastq
WIFL_Plate2_R1.fq WIFL_Plate2_R2.fq true 2
```

The "barcodes.txt" file here is a text file with the barcodes you used in your libraries, one per line. The last number (2) is the number of mismatches you are

allowing in your enzyme cutsite. In my tests, this did not make too much difference

4. Demultiplex and trim adapter sequences with stacks

```
mkdir demultiplexed

~/programs/stacks-1.39/process_radtags -i gzfastq -1
YWAR_Plate1_R1_2N.fq.gz -2 YWAR_Plate1_R2_2N.fq.gz -o
./demultiplexed -b YWAR1barcodes.txt -c -q -r -e sbfI --
filter_illumina --adapter_1 GATCGGAAGAGCACACGTCTGAACTCCAGTC --
adapter_2 CACTCTTTCCCTACACGACGCTCTTCCGATCT
```

Here are the options we're using:

-i file type (fastq.gz): Note that I compressed this file using "gzip filename"
-1 forward read file
-2 reverse read file
-o output directory
-b barcodes file: Different than in step 3 – see below!
-c clean data – remove uncalled bases
-q discard low quality reads
-r rescue barcodes
-e restriction enzyme
--filter_illumina remove reads that fail illumina chastity filters
--adapter_1 adapter to trim from forward read
--adapter_2 adapter to trim from reverse read

The barcodes file should be two columns: 1) the 8-letter barcodes and 2) the sample name. They should be separated by a tab with no header.

5. Remove PCR duplicates

Move orphaned reads into their own directory - we won't use these for now:

```
mkdir demultiplexed/orphans
mv demultiplexed/*rem* orphans/
mkdir dupfiltered
```

Remove duplicates

```
for sample in `ls *.fq.gz | cut -f1 -d'.'`
do
```

```
~/programs/stacks-1.39/clone_filter -1 $sample.1.fq.gz -2
$sample.2.fq.gz -i gzfastq -o ../dupfiltered
done
```

You can find the % clones in the stdout. Note that this job occasionally aborted when I requested only 4G of RAM. I have yet to run into a problem with 8G.

NOTE: My script for all of this is RADprocess.sh. You will need to change paths to your specific programs and files, but it should otherwise be fairly transferable