Usage: sl1p.pl <# of fofns> <list of fofns> <project name> [-r region] [-s seq info file] [-b barcode location] [-u threads] [-f force overwrite] [-q quality threshold] [-m chimera checking] [-p OTU picking algorithm] [-c OTU picking threshold] [-l linkage method] [-t taxonomic assignment method] [-d reference database] [-x timed]

input parameters
-r region of 16S [v3/v34/other] (default: v3)
-s seq info file to use with -r other (default: none)
-b location of barcode on sequence [fwd/rev] (default: fwd)
-u number of threads [0-9] (default: 1)
-f force overwrite output from previous runs [y/n] (default: n)
quality filtering
-q quality threshold [0-40] (default: 30)
-m chimera checking [y/n] (default: y)
OTU clustering
-p OTU picking algorithm [abundantotu/uclust/cdhit/dnaclust/uclust-ref/uclust-ref-strict/mothur/blast/uparse/all
-c OTU picking clustering threshold [0-100] (default: 97)
-l linkage method to use with -c mothur
taxonomic assignment
-t taxonomic assignment method [blast/rdp-training/all] (default: rdp-training)
-d reference database [gg2011/gg2013/silva111/all] (default: gg2011)
output parameters
-x output CPU timing of each data processing step [y/n] (default: n)

main()

assess command line options & usage
setup log & error files
check for necessary software install & version compatibility
ensure Illumina naming scheme
output software version info to log file
quality filtering
generate mapping file (perl)
trim away primers (cutadapt)
align paired-end reads (pandaseq)
cull any seq with Illumina seqing primers (cutadapt)
trim aligned read pairs based on $q (sickle)
adjust barcodes if necessary (perl)
consolidate fofn datasets (perl)
check mapping file (qiime)
if $m == yes
output read counts at each stage of qual filtering
(seq_log_info.txt)

OTU picking
if p == abundantOTU
p == uclust
p == cdhit
p == dnaclust
p == blast
p == mothur
p == uclust-ref
p == uclust-ref-strict
p == uparse
p == ALL

taxonomic assignment
for each combination of $clust & $gg
if t == blast
t == rdp-training
t == ALL

outputs
for each combination of $clust & $gg
align_seqs
filter_alignment
make_phylogeny
make_ggpruned
for each combination of $clust, $gg, & $taxon
make_otu_table
filter_otus
removeRoot
summarize_taxa
alpha_div
beta_div
RAnalysis
cleanUp

getPrimers()
if v3, v34
set hardcoded vars
else read in from file

return (set fwd, rev, min, & max length)

chimera_check()
identify_chimeric_seqs.py with -m usearch61 union between ref & de novo against d

filter chimeric seqs from fna input

abundantOTU()
AbundantOTU+ -d $dis

perl helper abundantOTU_to_qiime to format

uclust()
pick_otus.py -m uclust -s $dis

pick_rep_set.py -m most_abundant

cdhit()
pick_otus.py -m cdhit -s $dis

pick_rep_set.py -m most_abundant

dnaclust()
dnaclust -s $dis

perl helper editDnaclust-ForQiime.pl to format

pick_rep_set.py -m most_abundant

blast()
pick_otus.py -m uclust -s $dis

pick_rep_set.py -m most_abundant

qiime_mothur()
align_seqs.py

pick_otus.py -m mothur

pick_rep_set.py -m most_abundant

uclust_ref()
pick_otus.py -m uclust_ref

pick_rep_set.py -m most_abundant

uclust_ref_strict()
pick_otus.py -m uclust_ref -C

pick_rep_set.py -m most_abundant

uparse()
usearch -derep_fulllength
usearch -sortbysize
usearch -cluster_otus
usearch -usearch_global
perl helper script edit-UPARSEForQiime for format

taxa_blast()
assign_taxonomy.py -m blast -r $db -t $db

taxa_rdp()
assign_taxonomy.py -m rdp -c 0.5 -r $db -t $db

align_seq()
align_seqs.py -m pynast -a uclust -p 0.75

filter_alignment()
filter_alignment.py

make_phylogeny()
make_phylogeny()

make_ggpruned()
-perl to calculate all taxa in dataset
-prune $db phylogeny to -appropriate taxa format branch names

make_otu_table()
make_otu_table.py

filter_otus()
-use filter_otus_from_otu_table.py to filter singletons (-n 2) and doubletons (-n 3) into appropriately named output otu tables.
-biom convert to create .txt versions

removeRoot()
-perl auxilary script to remove any otu not assigned to bacteria from .txt
-biom convert to create .biom version

summarize_taxa()
summarize_taxa_through_plots.py

alpha_div()
alpha_rarefaction.py

beta_div()
-calculate the min read count in dataset
-uses pruned tree if produced (gg2011/13); otherwise uses rep_phylo.tre
-beta_diversity_through_plots.py -e min

RAnalysis()
-call to R to generate Rmd with read counts, taxa summarizes, alpha- & beta-diversity outputs
-create R output for user to start analysis

cleanUp()
-delete intermediary files
-compress important intermediaries

output:
log_seq_info.txt
log_sl1p_<Datetime>.err
.log
map_aux.tar.gz
map_<Proj>.txt
pandaseq_logs/
picked_otus_<clust>_<db>/
align_<db>_filtered.tar.gz
align_<db>.tar.gz
<gg>_<dis>_pruned.tre
library_stats_<taxon>_<gg>_n1_noRoot.txt
library_stats_<taxon>_<gg>.txt
otus_aux.tar.gz
otu_table_<taxon>_<gg>.biom
otu_table_<taxon>_<gg>_n1_noRoot.biom
otu_table_<taxon>_<gg>_n1_noRoot_relAbund.txt
otu_table_<taxon>_<gg>_n1_noRoot.txt
otu_table_<taxon>_<gg>.txt
phylo_aux.tar.gz
<taxon>_<db>_assigned_taxonomy_tr.tar.gz
rep_set_aux.tar.gz
seqs_<db>_otus.txt.tar.gz
sl1p_analysis.html
sl1p_analysis.Rmd
wf_taxa_summary_<taxon>_<db>
wf_taxa_summary_<taxon>_<db>_n1_noRoot
<Proj>.fna.tar.gz
qiime_params.txt
splits_dir.tar.gz