C_HW2: Makefile, command line exercises with yeast Version 8

Ken Youens-Clark

Abstract

Yeast is a well-characterized genome due to its small size and historical significance in genetics. The website http://yeastgenome.org/ is a dedicated resource for yeast genomics.

For this exercise, I want you to create a "Makefile" that will execute this entire pipeline if I type "make."

Citation: Ken Youens-Clark C_HW2: Makefile, command line exercises with yeast. **protocols.io** dx.doi.org/10.17504/protocols.io.ft9bnr6
Published: 16 Sep 2016

Protocol

Step 1.

If needed, clone the "metagenomics-book":

```
$ git clone https://github.com/kyclark/metagenomics-book.git
```

If you already have it, do "git pull" in that directory so that you can see "problems/yeast". Copy that into your "problems" directory.

```
$ cp -r metagenomics-book/problems/yeast abe487/problems
```

If you execute "make" you should see "echo OK" many times. Your job is to fill in the commands for all the targets.

Step 2.

'fasta' target:

This is an open access protocol distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited

Download all the '.fsa' files (chr 1-16, mt)
from http://downloads.yeastgenome.org/sequence/S288C_reference/chromosomes/fasta/ into a 'fasta' directory.

HINT: You can right-click on the links to copy the link location and then 'wget' the file.

Step 3.

"genome" target:

Make a single whole genome file called 'fasta/genome.fa'

Step 4.

"chr-count" target:

Count the chromosomes in the whole genome file. Put the number into a file called 'chr-count.'

HINT: Each of the original FASTA files contains a single chromosome.

Step 5.

"chr-size" target:

Find size of total genome. Put the answer into a file called 'chr-size.'

HINT: Look up the command 'wc' and find out what it does. The size of the genome can be determined by counting the number of characters in the genome (not on the same line as a fasta header).

Step 6.

"features" target:

Download the list of cerevisiae chromosome features:
http://downloads.yeastgenome.org/curation/chromosomal_feature/SGD_features.tab

This is an open access protocol distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited

Columns:

1. Primary Standfor Gene Database ID (SGDID) (mandatory)
2. Feature type (mandatory)
3. Feature qualifier (optional)
4. Feature name (optional)
5. Standard gene name (optional)
6. Alias (optional, multiples separated by |)
7. Parent feature name (optional)
8. Secondary SGDID (optional, multiples separated by |)
9. Chromosome (optional)1
10. Start_coordinate (optional)1
11. Stop_coordinate (optional)1
12. Strand (optional)1
13. Genetic position (optional)
14. Coordinate version (optional)
15. Sequence version (optional)
16. Description (optional)

Step 7.

'gene-count' target:

Count total genes ('ORF's)from "SGD_features.tab" into a file called 'gene-count.'

Step 8.

'verified-genes' target:

Count only verified genes from "SGD_features.tab" into a file called 'verified-genes.'

Step 9.

'uncharacterized-genes' target:

Count only uncharacterized genesfrom "SGD_features.tab" into a file called 'uncharacterized-genes.'

Step 10.

'palinsreg,' 'terminated-genes' targets:

This is an open access protocol distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited

Download the file 'ftp://ftp.imicrobe.us/abe487/yeast/palinsreg.txt'

1. These are detected terminator sequences in the *E. coli* genome (using the program GeSTer, if you're curious).
2. The command **grep '/G=[^]*' somefile** will find all lines that match /G=*somegenename*, where somegenename is a sequence of non-blank characters. Read the output of **man grep** and figure out how to -*o*nly print /G=*somegenename*, rather than the whole line.
3. Pipe the results of part (2) through a **cut** command to get only everything after the '='
4. Store the **unique, sorted** results of part (3) into a file named 'terminated_genes'

Step 11.

Add your Makefile and any other needed files to your Git repo and push it.

DO NOT ADD ANYTHING ELSE (e.g., the FASTA files)!!!

```
$ git add Makefile

$ git commit -m 'adding Makefile for C_HW2' Makefile

$ git push
```

You may notice that there is a ".gitignore" file in there that lists files that Git should ... well, ignore:

```
$ cat -n .gitignore
  1 fasta
  2 SGD_features.tab
  3 palinsreg.txt
```

Confirm that you can see 'problems/yeast/Makefile' in your web browser when you look at your Github repo. I will 'git pull' this, execute 'make,' and run the test to check that all the files exist and have the correct answers.

This is an open access protocol distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited