PYTHON EXAM **25 NOV 2016**

Description
Whenever you are predicting something, it is a good idea to evaluate the performance of the predictor. In the context of genome annotation, it would be relevant to know how good a gene predictor performs with respect to a known reference annotation.
Today you receive the sequence of yeast chromosome 3. You will annotate this sequence using the tool **`augustus`** and compare your annotation to the reference annotation of this chromosome. You need to compute the recall and precision (as defined below) and output some information on the correctly predicted transcripts.

Input
- The reference sequence of chromosome 3 of yeast (yeastchr3.fa)
- The reference annotation of yeast chromosome 3 (yeastchr3.gff)

Login to **`altschul.bioinformatics.nl`** and, on the command line, use the command **`wget`** to download the files from this location:
`http://www.bioinformatics.nl/courses/BIF-30806/docs/yeastchr3.fa`
`http://www.bioinformatics.nl/courses/BIF-30806/docs/yeastchr3.gff`

Assignment
Write a python script that performs the following tasks:
1. annotate the chr3 sequence using **`augustus`**
2. compare the predicted transcripts to the reference transcripts
3. report the recall and precision of the prediction
4. report the correctly predicted transcripts with their corresponding gene ID (in a tab-delimited file)

You could consider building the following functionality:
- Read filenames for both the reference sequence and the reference annotation from the command line (using argv)
- In your python script, run the program **`augustus`** to annotate the chr3 sequence (yeastchr3.fa).
    - Make sure augustus only predicts complete genes
    - Choose saccharomyces_cerevisiae_S288C as species (--species=...)
    - Call the output file **`augustus.gff`**
- Parse the GFF files to extract the relevant transcript information. Transcripts in the reference GFF are called “mRNA”; in the augustus output they are called “transcript”. In the 9th column of the reference transcripts, the gene responsible for the transcript (called Parent) is mentioned (e.g. `Parent=YCL069W`). See the GFF3 format specification: http://gmod.org/wiki/GFF3
- Calculate and report the recall and precision of the predicted transcripts. A transcript is correctly predicted (True Positive) if the start AND stop position are identical to those of a transcript in the reference annotation (overlap is not sufficient). Recall is the fraction of transcripts in the reference, which were correctly predicted by augustus (TruePositives/(TruePositives+FaseNegatives)). Precision is the fraction of predicted transcripts that is correct (TruePositives/ (TruePositives+FalsePositives)).

	In reference	*Not in reference*
Predicted	TruePositive	FalsePositive
Not predicted	FalseNegative	TrueNegative

- For each True Positive report [Start, Stop, AugID, GeneID] in a tab-delimited file, sorted on Start coordinate.

Output
The output of your script should look like this: (Note that the numbers in this output are made up. The numbers in your output will be different!)

On screen:
```
Recall: 0.72
Precision: 0.84
```

In the output file (tab-delimited):
```
Start   Stop    AugID   GeneID
...
25865   27616   g22.t1  YCL051W
37833   38801   g23.t1  YCL050C
39786   40724   g24.t1  YCL049C
41488   41730   g25.t1  YCL048W-A
...
```

Development
For development purposes, you can download this file:
`http://www.bioinformatics.nl/courses/BIF-30806/docs/tinyaugustus.gff`

Environment
You should work on the **altschul.bioinformatics.nl** server or on the **myers.bioinformatics.nl** server (if altschul is slow).
The program **`augustus`** is installed there. Try it by typing **`augustus`** or **`augustus --help`** on the command line. You should see information on the usage and options.

Additional notes
- Put your full name and student number as a comment in your script and put your username in the file name of your script.
- You may use the slides and the code from your exercises from the first course weeks. You cannot use BioPython or comparable packages, you should write your own parsers. You may not directly copy code from the Internet, but you may use it for inspiration.
- Running **`augustus`** takes some time. To avoid running it over and over again, make sure your code checks whether the output file exists.
- Think about your code organization (use subroutines).
- Document your code and make sure it has good style and readability.
- Make sure you hand in a working script. If it is unfinished, you can leave the unfinished part in comments.

How to hand it in?
On Friday 25-11-2016 before 12.15, submit your Python script in BlackBoard under "20161125_exam"

Assessment
We will run your script on the input, check the output, and assess the quality of your code. The grade for this assignment will be 40% of your course grade.