How to install and execute the trimmmomatic package

Henry R. Moncada

November 10, 2018

Contents

1 Import Modules in python

1. Pip - A cross-platform package manager for installing and managing `Python 2 >=2.7.9` or `Python 3 >=3.4` software packages

```
$ sudo apt-get install python-pip
or
$ sudo apt-get install python-pip3
```

2. Numpy - Linear algebra library, General-purpose linear algebra, array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

   - Using `sudo apt-get`

     ```
     $ sudo apt-get install python-numpy
     ```

   - Using `pip`

     ```
     $ pip install --U numpy
     ```

3. Panda - Data processing and analysis library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

   - Using `sudo apt-get`

     ```
     $ sudo apt-get install python-pandas
     ```

   - Using `pip`

     ```
     $ pip install --U pandas
     ```

4. Re - Regular expression operations, this module provides regular expression matching operations similar to those found in Perl.

```
$ pip install --U scikit-learn
```

```
import re

# Finding Patterns in Text:
"""
The most common use for re is to search for patterns in text.
This example looks for two literal strings, 'this' and 'that', in a text string.
"""
patterns = [ 'this', 'that' ]
text = 'Does this text match the pattern?'
"""
search() takes the pattern and text to scan, and returns a Match object when the pattern is found.
If the pattern is not found, search() returns None.
"""
for pattern in patterns:
    print 'Looking for "%s" in "%s" ->' % (pattern, text),

    if re.search(pattern,  text):
        print 'found a match!'
    else:
        print 'no match'

"""
The Match object returned by search() holds information about the nature of the match,
including the original input string, the regular expression used, and the location within
the original string where the pattern occurs.
"""

pattern = 'this'
text = 'Does this text match the pattern?'

match = re.search(pattern, text)

"""
The start() and end() methods give the integer indexes into the string showing where the text matched by the pattern occurs.
"""
s = match.start()
e = match.end()

print 'Found "%s" in "%s" from %d to %d ("%s")' % \
    (match.re.pattern, match.string, s, e, text[s:e])
```

5. Scikit-learn (formerly scikits.learn) is a free software machine learning library for the Python programming language.

```
$ pip install --U scikit-learn
```

6. NLTK - Natural Language Toolkit, leading platform for building Python programs to work with human language data.

   - Open linux terminal
   - Install using `sudo apt-get`

     ```
     $ sudo apt-get install python-nltk
     ```

   - Call Python

     ```
     $ python
     Python 2.7.12 (default, Nov 19 2016, 06:48:10)
     [GCC 5.4.0 20160609] on linux2
     Type "help", "copyright", "credits" or "license" for more information.
     >>>
     ```

   - Import and download nltk library packages

     ```
     >>> import nltk
     >>> nltk.download()
     showing info http://www.nltk.org/nltk_data/
     ```

7. A windows will open click and install `all`. Test the package installation with the following program

```
# Load package
import nltk

# Sentences
sentence = """At eight o'clock on Thursday morning Arthur didn't feel very good."""

# Sentence Tokenization
tokens = nltk.word_tokenize(sentence)
print tokens

# POS Tagging
tagged = nltk.pos_tag(tokens)
print tagged[0:6]

# (NE Chunking) Identify named entities:
entities = nltk.chunk.ne_chunk(tagged)
print entities

# Treebank Corpus
from nltk.corpus import treebank
t = treebank.parsed_sents('wsj_0001.mrg')[0]
print t

# Draw
print "Draw"
t.draw()
```

8. Biopython - A set of freely available tools for biological computation written in Python by an international team of developers.

   - Open linux terminal
   - Install using `sudo apt-get`

   ```
   $ sudo apt-get install python-biopython


   $ sudo apt-get install python3-biopython
   ```

9. SciPy - An open-source Python library used for scientific computing and technical computing. It contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering.

   ```
   $ pip install --U scipy
   ```

2 FASTQ Format

FASTQ files are storage text files for DNA sequence data with a quality (Phred) score for each base. The sequence letter and the quality score are represented with ASCII character. Each sequence gets four lines in the file: one for sequence identifier (start with "@"), one for the nucleotide sequence, one description line (often just a "+" character), and last one for the per-base quality scores (thus lines two and four must be equal length).

2.1 Format

An example of a valid entry is as follows; note the space preceding the read number element:

```
@SIM:1:FCX:1:15:6329:1045 1:N:0:2
TCGCACTCAACGCCCTGCATATGACAAGACAGAATC
+
<>;##=><9=AAAAAAAAAA9#:<#<;<<<????#=
```

Each entry in a FASTQ file consists of four lines:

- Line 1 (Label): Begins with a "@" character and is followed by a sequence identifier

```
@     SIM    :     1      :      FCX    :  1   : 15   : 6329  : 1045      1  :      N      :     0          :       2

@<instrument>:<run number>:<flowcell ID>:<lane>:<tile>:<x-pos>:<y-pos> <read>:<is filtered>:<control number>: <sample number>
```

- Line 2 (Sequence): Sequence letters

  `TCGCACTCAACGCCCTGCATATGACAAGACAGAATC`

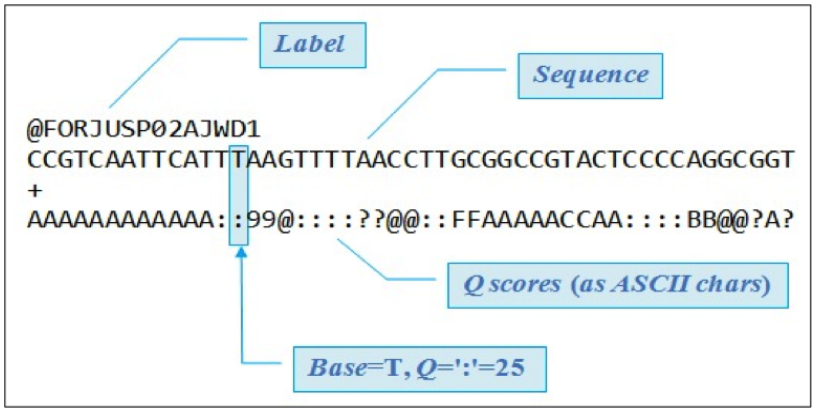
- Line 3: Quality score identifier line (consisting only of a "+")

  `+`

- Line 4 (Q scores): Quality score, encodes the quality values for the sequence in Line 2

  `<>;##=><9=AAAAAAAAAA9#:<#<;<<<????#=`

  – NOTE: For the undetermined FASTQ files only, the sequence observed in the index read is written to the FASTQ header in place of the sample number. This information can be useful for troubleshooting demultiplexing.



Fasq format

The following table describes the elements of Line 1 (See Table 1)

`@<instrument>:<run number>:<flowcell ID>:<lane>:<tile>:<x-pos>:<y-pos> <read>:<is filtered>:<control number>: <sample number>`

Fastq Format- Sequence identifier

Element	Requirements	Description
@	@	Each sequence identifier line starts with @
< instrument >	Characters allowed: a-z, A-Z, 0-9 and underscore	Instrument ID
< run number >	Numerical	
< flowcell ID >	Characters allowed: a-z, A-Z, 0-9	Run number on instrument
< lane >	Numerical	Lane number
< tile >	Numerical	Tile number
< x_pos >	Numerical X	Coordinate of cluster
< y_pos >	Numerical Y	Coordinate of cluster
< read >	Numerical	Read number. 1 can be single read or Read 2 of paired-end
< is filtered >	Y or N	Y if the read is filtered (did not pass), N otherwise
< control number >	Numerical	0 when none of the control bits are on, otherwise it is an even number. On HiSeq X and NextSeq systems, control specification is not performed and this number is always 0.
< sample number >	Numerical	Sample number from sample sheet

3 Trimmomatic

Trimmomatic is a lightweight and multithreaded command line java tool application that can be used to

- To trim and crop Illumina FASTQ data.
- To remove Illumina adapter sequences and low quality reads.
- To performs a variety of useful trimming tasks for illumina paired-end and single end mode. For single-ended data, one input and one output file are specified, plus the processing steps. For paired-end data, two input files are specified, and 4 output files, 2 for the `paired` output where both reads survived the processing, and 2 for corresponding `unpaired` output where a read survived, but the partner read did not.
- It works with FASTQ files (using phred + 33 or phred + 64 quality scores, depending on the Illumina pipeline used).
- It uses a sliding window to analyze chunks of each read, examining the quality score, minimum read length, if it corresponds to an adapter sequence, etc.
- Files compressed such as `gzip` or `bzip2` are supported, and are identified by use of `.gz` or `.bz2` file extensions.

The selection of trimming steps and their associated parameters are supplied on the command line.

- ILLUMINACLIP: Cut adapter and other illumina-specific sequences from the read.
- SLIDINGWINDOW: Perform a sliding window trimming, cutting once the average quality within the window falls below a threshold.
- LEADING: Cut bases off the start of a read, if below a threshold quality
- TRAILING: Cut bases off the end of a read, if below a threshold quality
- CROP: Cut the read to a specified length
- HEADCROP: Cut the specified number of bases from the start of the read
- MINLEN: Drop the read if it is below a specified length
- TOPHRED33: Convert quality scores to Phred-33
- TOPHRED64: Convert quality scores to Phred-64

4 Install Trimmmomatic

- Download `trimmomatic_0.38`

```
$ wget http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/Trimmomatic-Src-0.38.zip
```

- Extract the zip file

```
$ unzip Trimmomatic-Src-0.38.zip
```

- Rename the folder as `TRIMHOME`

```
$ mv Trimmomatic-Src-0.38 TRIMHOME
$ cd TRIMHOME
```

- Show the folder

```
henry@Lola:~/TRIMHOME$ tree -d
.
|-- adapters
|-- classes
    |-- demo
    |-- META-INF
    |-- org
        |-- itadaki
            |-- bzip2
        |-- usadellab
            |-- trimmomatic
                |-- fasta
                |-- fastq
                |-- threading
                |-- trim
                |-- util
|-- data
    |-- paired_end
    |-- single_end
|-- distSrc
|-- lib
|-- src
    |-- org
        |-- usadellab
            |-- trimmomatic
                |-- fasta
                |-- fastq
                |-- threading
                |-- trim
                |-- util

28 directories
```

- TREE `TRIMHOME/adapters/`

```
henry@Lola:~/TRIMHOME$ tree adapters
adapters
|-- NexteraPE-PE.fa
|-- TruSeq2-PE.fa
|-- TruSeq2-SE.fa
|-- TruSeq3-PE-2.fa
|-- TruSeq3-PE.fa
|-- TruSeq3-SE.fa
```

- Check if you are really the superuser

```
henry@Lola:~$ $ whoami
henry
```

- The makefile will build

  1. The `java jar` file `trimmmomatic.jar`,
  2. The `TRIMHOME/classes` folder
  3. Copy all remain folders from `TRIMHOME`

  Use the following makefile to help you with the installation.

- Makefile

```
# http://semver.org/
VERSION := 0.38.0
INSTALL := ${HOME}   # /home/henry
SOURCES := $(shell find src/ -name "*.java" | sed 's|src/||g')

all:
rm -rf classes; mkdir -p classes/; \
cd src/; javac -classpath ".:../lib/jbzip2-0.9.jar" -d ../classes ${SOURCES}; \
cd ../classes; jar xf ../lib/jbzip2-0.9.jar; \
jar cmf ../MANIFEST.MF trimmomatic.jar ./

check:
@echo "this package doesn't have any test (yet)"

install:
mkdir -p ${INSTALL}/TRIMHOME                        # Created folder /home/henry/TRIMHOME
cp classes/trimmomatic.jar ${INSTALL}/TRIMHOME/     # Copy  classes/trimmomatic.jar into  /home/henry/TRIMHOME/trimmomatic.jar
cp -vrf * ${INSTALL}/TRIMHOME/                      # Copy  all filestrimmomatic-038

dist:
mkdir -p trimmomatic-${VERSION}
cp AUTHORS Makefile README trimmomatic-${VERSION}
cp build.xml MANIFEST.MF versionHistory.txt trimmomatic-${VERSION}
cp -r adapters distSrc lib/ src/ trimmomatic-${VERSION}
tar -czvf trimmomatic-${VERSION}.tar.gz trimmomatic-${VERSION}/
rm -rf trimmomatic-${VERSION}

clean:
rm -rf classes/
```

- Use Makefile

1. Install `trimmomatic-0.38` packages without know where (No recommemded)

```
henry@Lola:~$ cd trimmomatic-0.38

henry@Lola:~/trimmomatic-0.38$  make install
```

   - Find `$HOME`,

```
henry@Lola:~$ $HOME
bash: /home/henry: Is a directory
```

   - Find where was installed

```
henry@Lola:~/TRIMHOME$ pwd
/home/henry/TRIMHOME
```

2. Install `trimmomatic-0.38` packages on a specific location
   - Where

```
/usr/local/TRIMHOME/
```

   - How we installed the trimmomatic packages

```
make
make check
make install INSTALL="/usr/local/"
```

- TREE data

```
henry@Lola:~/data$ tree data
data
|
|-- paired_end
    |-- ERR950159_1.fastq
    |-- ERR950159_2.fastq
    |-- ERR950173_1.fastq
    |-- ERR950173_2.fastq
    |-- ERR950179_1.fastq
    |-- ERR950179_2.fastq
|
|-- single_end
    |-- ERR458493.fastq
    |-- ERR458493.fastq.gz
    |-- ERR458494.fastq.gz
    |-- ERR458495.fastq.gz
    |-- SRR020192.fastq.gz
    |-- yeast_data.tar.gz
```

5 How Trimmomatic Work

- Trimmomatic is a program written in the Java programming language.

1. Check your Java version

```
$ java -version
openjdk version "1.8.0_131"
OpenJDK Runtime Environment (build 1.8.0_131-8u131-b11-0ubuntu1.16.04.2-b11)
OpenJDK 64-Bit Server VM (build 25.131-b11, mixed mode)
```

   or

```
$ javac -version
javac 1.8.0_131
```

- Input files (single end and paired end)

```
henry@Lola:~/TRIMHOME$$ tree data
data
|
|-- paired_end
    |-- ERR950159_1.fastq
    |-- ERR950159_2.fastq
    |-- ERR950173_1.fastq
    |-- ERR950173_2.fastq
    |-- ERR950179_1.fastq
    |-- ERR950179_2.fastq
|
|-- single_end
    |-- ERR458493.fastq
    |-- ERR458493.fastq.gz
    |-- ERR458494.fastq.gz
    |-- ERR458495.fastq.gz
    |-- SRR020192.fastq.gz
    |-- yeast_data.tar.gz

2 directories, 23 files
```

- Output files

```
henry@Lola:~/TRIMHOME$$ tree data
data
|
|-- paired_end
    |-- output_forward_paired_ERR950159_1.fastq
    |-- output_forward_unpaired_ERR950159_1.fastq
    |-- output_reverse_paired_ERR950159_2.fastq
    |-- output_reverse_unpaired_ERR950159_2.fastq
|
|-- single_end
    |-- OUT_ERR458493.fastq


2 directories, 23 files
```

- The complete command for Trimmomatic
  1. HOME

     ```
     $ $HOME
     bash: /home/henry: Is a directory
     ```

  2. Single end

     ```
     $ java -jar /usr/local/TRIMHOME/trimmomatic.jar SE -threads 4 -phred33
     ~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/single_end/ERR458493.fastq
     ~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/single_end/OUT_ERR458493.fastq
     ILLUMINACLIP:/usr/local/TRIMHOME/adapters/TruSeq3-SE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36

     TrimmomaticSE: Started with arguments:
      -threads 4 -phred33 /home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/single_end/ERR458493.fastq
      /home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/single_end/OUT_ERR458493.fastq
      ILLUMINACLIP:/usr/local/TRIMHOME/adapters/TruSeq3-SE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36
     Using Long Clipping Sequence: 'AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTA'
     Using Long Clipping Sequence: 'AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC'
     ILLUMINACLIP: Using 0 prefix pairs, 2 forward/reverse sequences, 0 forward only sequences, 0 reverse only sequences
     Input Reads: 1093957 Surviving: 1073395 (98.12%) Dropped: 20562 (1.88%)
     TrimmomaticSE: Completed successfully
     ```

  3. Paired end

     ```
     $ java -jar /usr/local/TRIMHOME/trimmomatic.jar PE -threads 12 -phred33
     ~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/ERR950159_1.fastq
     ~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/ERR950159_2.fastq
     ~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_forward_paired_ERR950159_1.fastq
     ~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_forward_unpaired_ERR950159_1.fastq
     ~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_reverse_paired_ERR950159_2.fastq
     ~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_reverse_unpaired_ERR950159_2.fastq
     ILLUMINACLIP:/usr/local/TRIMHOME/adapters/TruSeq3-PE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36

     TrimmomaticPE: Started with arguments:
      -threads 12 -phred33 /home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/ERR950159_1.fastq
      /home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/ERR950159_2.fastq
      /home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_forward_paired_ERR950159_1.fastq
      /home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_forward_unpaired_ERR950159_1.fastq
      /home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_reverse_paired_ERR950159_2.fastq
      /home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_reverse_unpaired_ERR950159_2.fastq
      ILLUMINACLIP:/usr/local/TRIMHOME/adapters/TruSeq3-PE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36
     Using PrefixPair: 'TACACTCTTTCCCTACACGACGCTCTTCCGATCT' and 'GTGACTGGAGTTCAGACGTGTGCTCTTCCGATCT'
     ILLUMINACLIP: Using 1 prefix pairs, 0 forward/reverse sequences, 0 forward only sequences, 0 reverse only sequences
     Input Read Pairs: 21433074 Both Surviving: 17241380 (80.44%) Forward Only Surviving: 3912653 (18.26%) Reverse Only Surviving: 143989 (0.67%) Dropped: 135052 (0.63%)
     TrimmomaticPE: Completed successfully
     ```

- Let break the command to run Trimmomatic
  1. Trimommatic starts with

     ```
     $ java -jar /path_to_the_java_jar/trimmomatic.jar
     ```

     - `java` tells our computer that were running a Java program.
     - `-jar` is an option specifying that were going to specify the location of the Java program we want to run
     - `/path_to_the_java_jar/trimmomatic.jar` is the java executable tool.

  2. Trimommatic attributes
     - `PE`
     - `-threads 12`
     - `-phred33`
  3. Input file
     (a) Single end

     ```
     ~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/single_end/ERR458493.fastq
     ```

     (b) Paired end

     ```
     ~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/ERR950159_1.fastq
     ~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/ERR950159_2.fastq
     ```

  4. Output Files

(a) Single end

~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/single_end/OUT_ERR458493.fastq

(b) Paired end

~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_forward_paired_ERR950159_1.fastq
~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_forward_unpaired_ERR950159_1.fastq
~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_reverse_paired_ERR950159_2.fastq
~/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_reverse_unpaired_ERR950159_2.fastq

5. ILLUMINA Attributes

(a) Single end

ILLUMINACLIP:/usr/local/TRIMHOME/adapters/TruSeq3-SE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36

(b) Paired end

ILLUMINACLIP:/usr/local/TRIMHOME/adapters/TruSeq3-PE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36

6 Python code to execute trimmmomatic

- Example 1:

```
# Load modules
from Bio import SeqIO  # module needed for sequence input
import sys             # module needed for command line argument to get fastq name
import re              # module for Regular expression operations, this module provides regular expression matching operations similar to those found in Perl.
import os

# Compiler and path tool
ja = " java -jar"
tool = " /usr/local/TRIMHOME/trimmomatic.jar"
s = " "
# Execute Single end or Paired end
mode = raw_input("Please enter enter mode (Single [1]  or Paired [2]): ")
print 'Mode = ', mode

if mode == 'Single' or mode == '1':
   print "Sigle End mode : Supply fastq file\n"
   # Load data
   file_input = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/single_end/ERR458493.fastq"
   fastqfile1 = open(file_input, 'r')
   # Output single end
   file_output = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/single_end/output_single_ERR458493.fastq"
   # FASTQ file where you want to trim all the reads
   count = 0
   for rec in SeqIO.parse(fastqfile1, "fastq"):
      count += 1
   print("farsq %i reads\n" % count)   # count reads

   print "Start Trimmonatics :"    #java -jar trimmomatic SE -threads 4 -phred33 file_input file_output ILLUMINACLIP:adapters:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36
   print "--------------------"
   attribute              = " SE -threads 4 -phred33"
   illuminaclip_adapters  = "ILLUMINACLIP:/usr/local/TRIMHOME/adapters/TruSeq3-SE.fa:2:30:10"
   illuminaclip_Attribute = "LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36"
   cmd  = s + ja + s + tool + s + attribute + s + file_input + s + file_output + s + illuminaclip_adapters + s + illuminaclip_Attribute
   os.system(cmd)

elif mode == 'Paired' or mode == '2':
   print "Paired End mode: Supply a forward and reverse fastq files\n"
# Load data forward
   filename_forward = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/ERR950159_1.fastq"
   fastq_forward = open(filename_forward, 'r')  # Here you write your fastq filename
   # FASTQ file where you want to trim all the reads
   #count = 0
   #for rec in SeqIO.parse(fastq_forward, "fastq"):
   #   count += 1
   #print("fastq forward %i reads" % count)   # count reads

   # Load data reverse
   filename_reverse = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/ERR950159_2.fastq"
   fastq_reverse = open(filename_reverse, 'r')
   # FASTQ file where you want to trim all the reads
   #count = 0
   #for rec in SeqIO.parse(fastq_reverse, "fastq"):
   #   count += 1
   #print("fastq reverse %i reads" % count)   # count reads

   # Outputs paired end
   output_forward_paired = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_forward_paired_ERR950159_1.fastq"
   output_forward_unpaired = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_forward_unpaired_ERR950159_1.fastq"
   output_reverse_paired = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_reverse_paired_ERR950159_2.fastq"
   output_reverse_unpaired = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_reverse_unpaired_ERR950159_2.fastq"

   print "Start Trimmonatics :"
   print "--------------------"
   attribute              = "PE -threads 12 -phred33 "
   illuminaclip_adapters  = "ILLUMINACLIP:/usr/local/TRIMHOME/adapters/TruSeq3-PE.fa:2:30:10"
   illuminaclip_Attribute = "LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36"
   cmd1 = s + ja + s + tool + s + attribute + s + filename_forward + s + filename_reverse + s + output_forward_paired + s + output_forward_unpaired
   cmd2 = s + output_reverse_paired + s + output_reverse_unpaired + s + illuminaclip_adapters + s + illuminaclip_Attribute
   cmd  =  cmd1 + cmd2
   os.system(cmd)
```

- Example 2:

```
# Load modules
from Bio import SeqIO  # module needed for sequence input
import sys             # module needed for command line argument to get fastq name
import re              # module for Regular expression operations, this module provides regular expression matching operations similar to those found in Perl.
```

```
import subprocess

# Compiler and path tool
ja = " java -jar"
tool = " /usr/local/TRIMHOME/trimmomatic.jar"
s = " "
# Execute Single end or Paired end
mode = raw_input("Please enter enter mode (Single [1]  or Paired [2]): ")
print 'Mode = ', mode

if mode == 'Single' or mode == '1':
   print "Sigle End mode : Supply fastq file\n"
   # Load data
   file_input = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/single_end/ERR458493.fastq"
   fastqfile1 = open(file_input, 'r')
   # Output single end
   file_output = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/single_end/output_single_ERR458493.fastq"
   # FASTQ file where you want to trim all the reads
   count = 0
   for rec in SeqIO.parse(fastqfile1, "fastq"):
      count += 1
   print("farsq %i reads\n" % count)   # count reads

   print "Start Trimmonatics :"    #java -jar trimmomatic SE -threads 4 -phred33 file_input file_output ILLUMINACLIP:adapters:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36
   print "--------------------"
   attribute              = " SE -threads 4 -phred33"
   illuminaclip_adapters  = "ILLUMINACLIP:/usr/local/TRIMHOME/adapters/TruSeq3-SE.fa:2:30:10"
   illuminaclip_Attribute = "LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36"
   cmd  = s + ja + s + tool + s + attribute + s + file_input + s + file_output + s + illuminaclip_adapters + s + illuminaclip_Attribute
   call = ["/bin/bash", "-c", cmd]
   ret  = subprocess.call(call, stdout=None, stderr=None)
   if ret > 0:
      print "Warning - result was %d" % ret

elif mode == 'Paired' or mode == '2':
   print "Paired End mode: Supply a forward and reverse fastq files\n"
# Load data forward
   filename_forward = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/ERR950159_1.fastq"
   fastq_forward = open(filename_forward, 'r')  # Here you write your fastq filename
   # FASTQ file where you want to trim all the reads
   #count = 0
   #for rec in SeqIO.parse(fastq_forward, "fastq"):
   #   count += 1
   #print("fastq forward %i reads" % count)   # count reads

   # Load data reverse
   filename_reverse = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/ERR950159_2.fastq"
   fastq_reverse = open(filename_reverse, 'r')
   # FASTQ file where you want to trim all the reads
   #count = 0
   #for rec in SeqIO.parse(fastq_reverse, "fastq"):
   #   count += 1
   #print("fastq reverse %i reads" % count)   # count reads

   # Outputs paired end
   output_forward_paired = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_forward_paired_ERR950159_1.fastq"
   output_forward_unpaired = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_forward_unpaired_ERR950159_1.fastq"
   output_reverse_paired = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_reverse_paired_ERR950159_2.fastq"
   output_reverse_unpaired = "/home/henry/Desktop/RNA_Seq_Differential_Expressions/FASTQ_Files/paired_end/output_reverse_unpaired_ERR950159_2.fastq"

   print "Start Trimmonatics :"
   print "--------------------"
   attribute              = "PE -threads 12 -phred33 "
   illuminaclip_adapters  = "ILLUMINACLIP:/usr/local/TRIMHOME/adapters/TruSeq3-PE.fa:2:30:10"
   illuminaclip_Attribute = "LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36"
   cmd1 = s + ja + s + tool + s + attribute + s + filename_forward + s + filename_reverse + s + output_forward_paired + s + output_forward_unpaired
   cmd2 = s + output_reverse_paired + s + output_reverse_unpaired + s + illuminaclip_adapters + s + illuminaclip_Attribute
   cmd  =  cmd1 + cmd2
   call = ["/bin/bash", "-c", cmd]
   ret  = subprocess.call(call, stdout=None, stderr=None)
   if ret > 0:
      print "Warning - result was %d" % ret
```