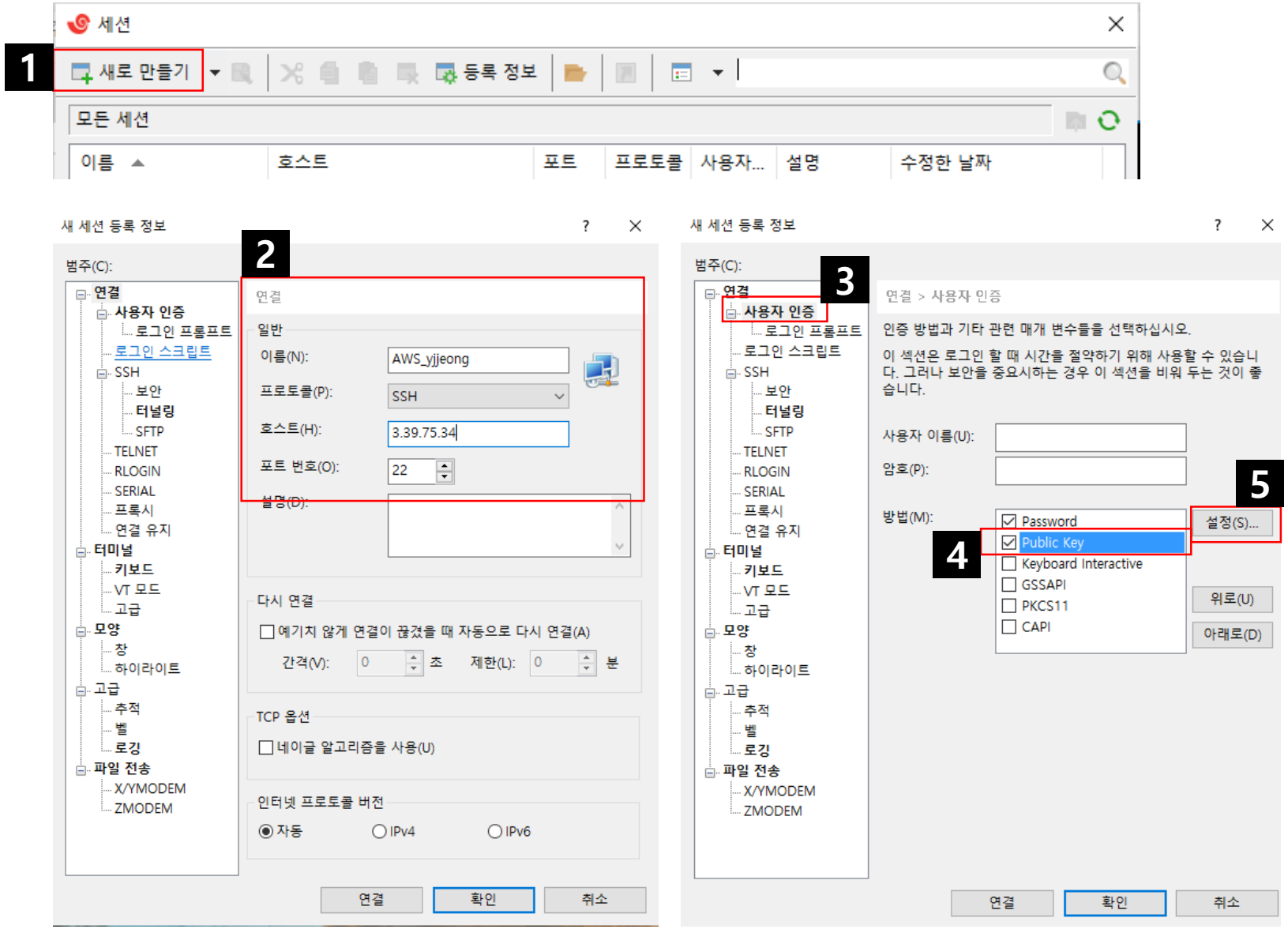
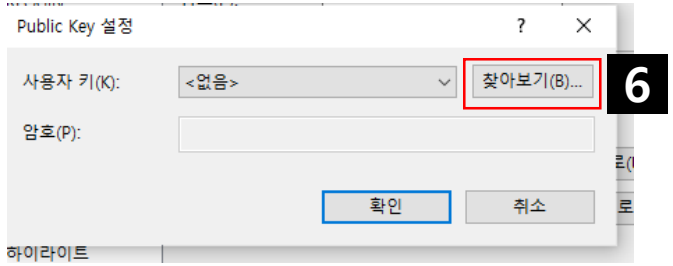
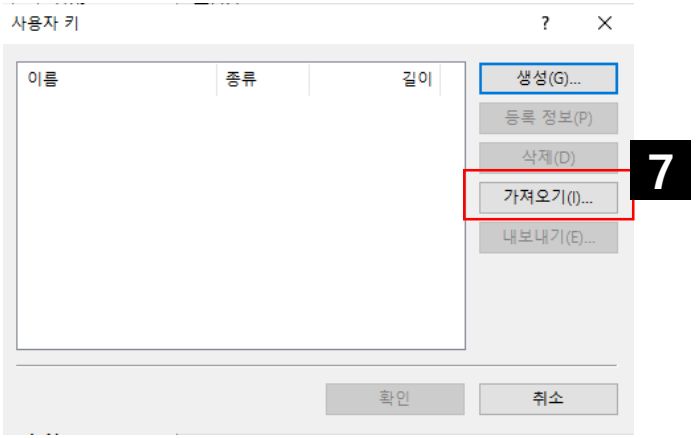
Lecture 8

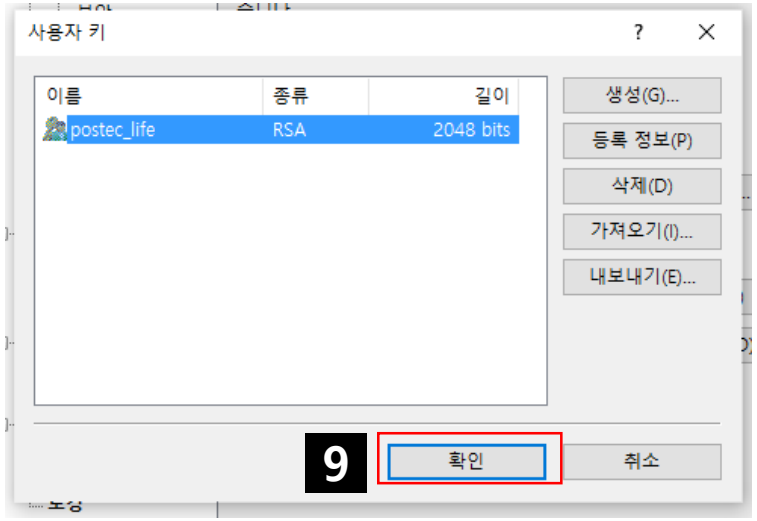
FASTA/FASTQ, Installing UNIX programs, Accessing SRA

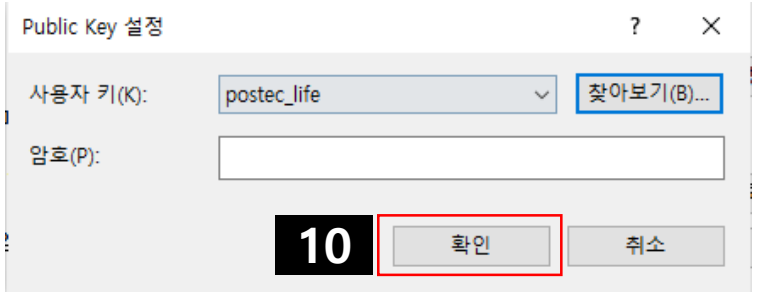
AWS 서버 접속 방법

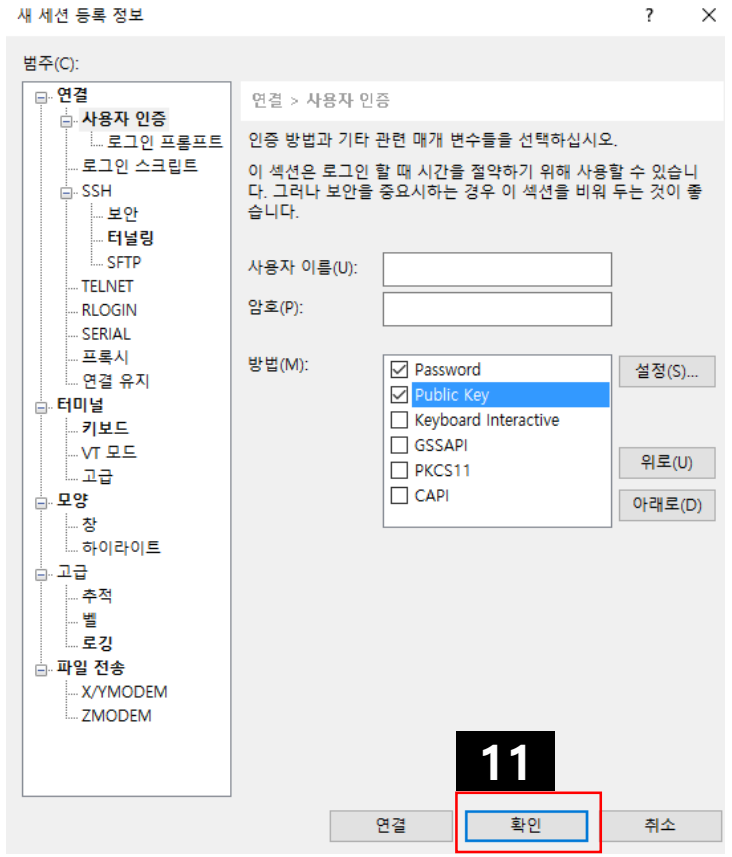

1
세션
새로 만들기
등록 정보
모든 세션
이름
호스트
포트
프로토콜
사용자...
설명
수정한 날짜
새 세션 등록 정보
2
범주(C):
연결
사용자 인증
로그인 프롬프트
로그인 스크립트
SSH
보안
터널링
SFTP
TELNET
RLOGIN
SERIAL
프록시
연결 유지
터미널
키보드
VT 모드
고급
모양
창
하이라이트
고급
추적
벨
로깅
파일 전송
X/YMODEM
ZMODEM
일반
이름(N):
AWS_yjjeong
프로토콜(P):
SSH
호스트(H):
3.39.75.34
포트 번호(O):
22
설명(D):
다시 연결
예기치 않게 연결이 끊겼을 때 자동으로 다시 연결(A)
간격(V):
초
제한(L):
분
TCP 옵션
네이글 알고리즘을 사용(U)
인터넷 프로토콜 버전
자동
IPv4
IPv6
연결
확인
취소
새 세션 등록 정보
3
범주(C):
연결 > 사용자 인증
인증 방법과 기타 관련 매개 변수들을 선택하십시오.
이 섹션은 로그인 할 때 시간을 절약하기 위해 사용할 수 있습니다. 그러나 보안을 중요시하는 경우 이 섹션을 비워 두는 것이 좋습니다.
사용자 이름(U):
암호(P):
방법(M):
4
Password
Public Key
Keyboard Interactive
GSSAPI
PKCS11
CAPI
5
설정(S)...
위로(U)
아래로(D)
연결
확인
취소



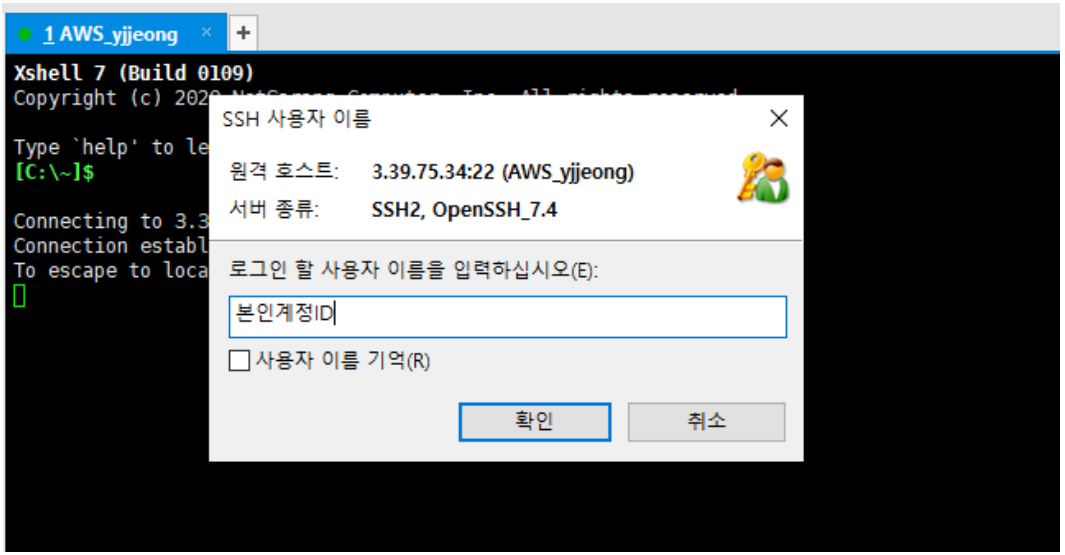


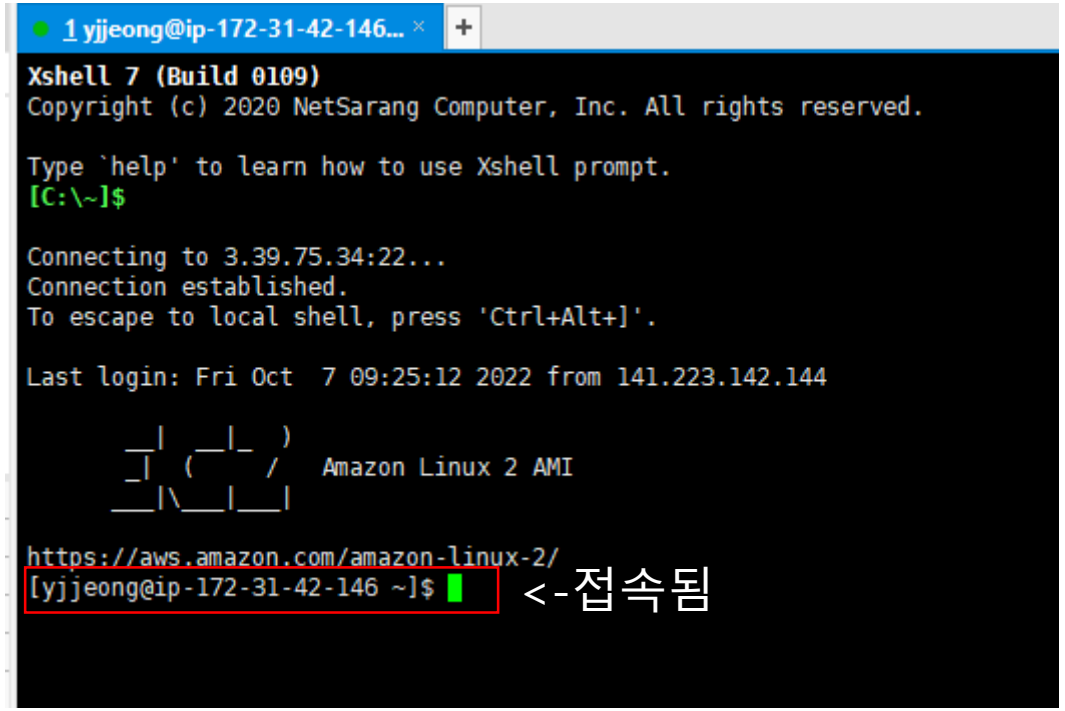




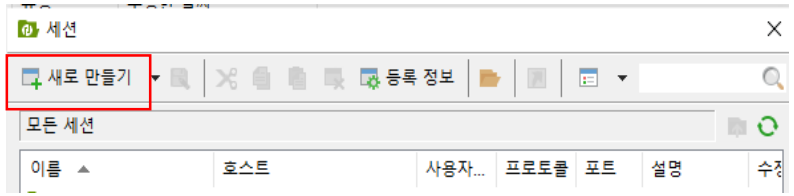


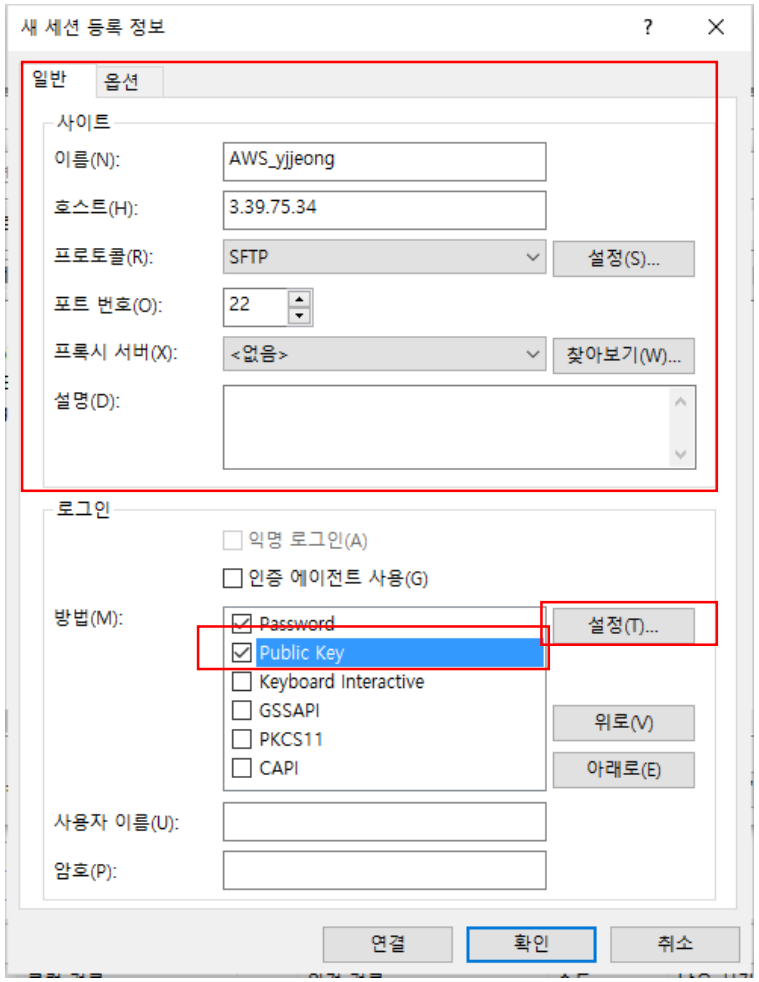
8 PLMS에서 받은 postec_life.pem 파일 선택

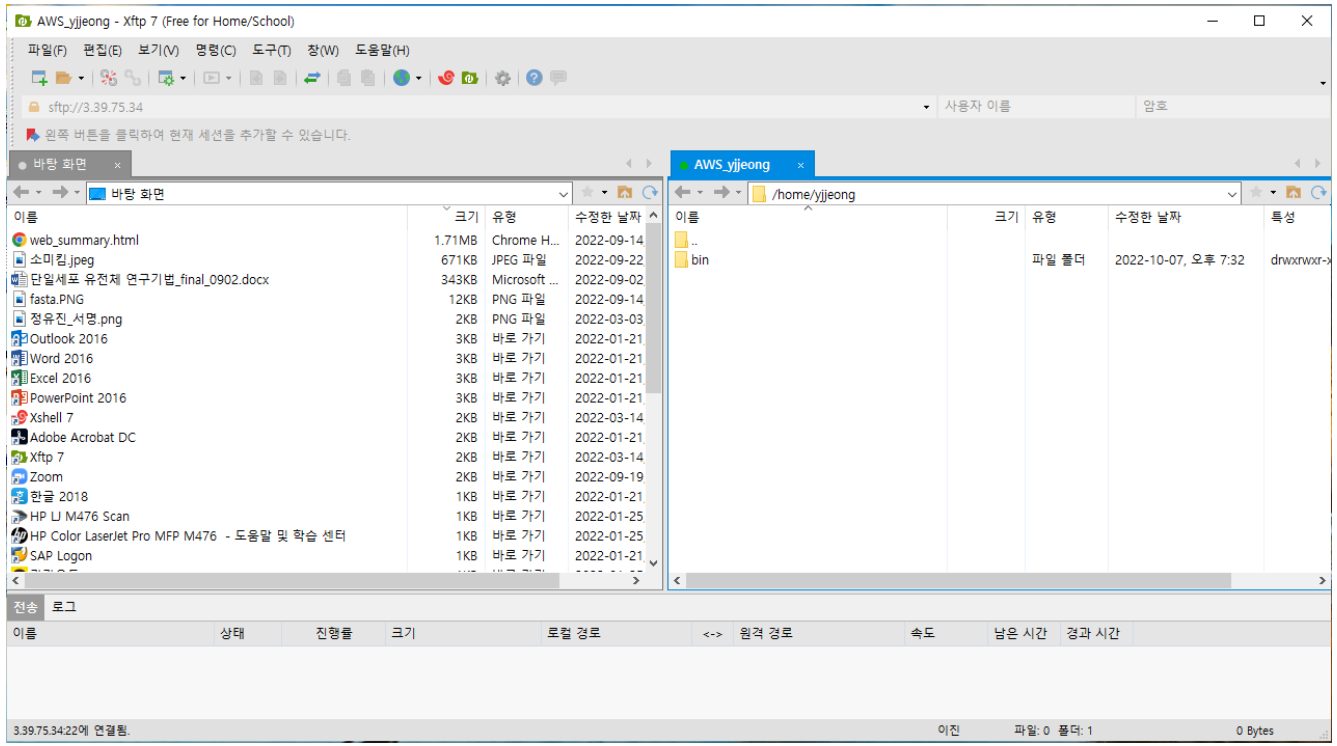

1 AWS_yjjeong
Xshell 7 (Build 0109)
Type `help' to le
[C:\~]$
Connecting to 3.3
Connection establ
To escape to loca
SSH 사용자 이름
원격 호스트: 3.39.75.34:22 (AWS_yjjeong)
서버 종류: SSH2, OpenSSH_7.4
로그인 할 사용자 이름을 입력하십시오(E):
본인계정ID
사용자 이름 기억(R)
확인
취소


1 yjjeong@ip-172-31-42-146...
Xshell 7 (Build 0109)
Copyright (c) 2020 NetSarang Computer, Inc. All rights reserved.
Type `help' to learn how to use Xshell prompt.
[C:\~]$
Connecting to 3.39.75.34:22...
Connection established.
To escape to local shell, press 'Ctrl+Alt+]'.
Last login: Fri Oct 7 09:25:12 2022 from 141.223.142.144
Amazon Linux 2 AMI
https://aws.amazon.com/amazon-linux-2/
[yjjeong@ip-172-31-42-146 ~]$
<-접속됨

Xftp에서 새 세션 등록 후 파일 업로드/다운로드 가능







FASTA/FASTQ

Overview

- Nucleotide (and protein) sequences are stored in two plain-text formats widespread in bioinformatics: FASTA and FASTQ.

- We will discuss each format and their limitations, and then see some tools for working with data in these formats.

FASTA

- The FASTA format originates from the FASTA alignment suites, created by William R. Pearson and David J. Lipman (https://www.ncbi.nlm.nih.gov/pubmed/2983426). The FASTA format is used to store any sort of sequence data not requiring per-base pair quality scores.

- This includes: reference genome files, protein sequences, coding DNA sequences (CDS), transcript sequences, and so on.

FASTA

- FASTA files are composed of sequence entries, each containing two parts: a description and the sequence data.

- The description line begins with a “greater than” symbol (>) and contains the sequence identifier and other optional information

- The sequence data begins on the next line after the description, and continues until there’s another description line.

FASTA

- An example FASTA file:

```
(base) head -n 10 /home/Data/egfr_flank.fasta
>ENSMUSG00000020122|ENSMUST00000138518
CCCTCCTATCATGCTGTCAGTGTATCTCTAAATAGCACTCTCAACCCCCGTGAACTTGGT
TATTAAAAACATGCCCAAAGTCTGGGAGCCAGGGCTGCAGGGAAATACCACAGCCTCAGT
TCATCAAAACAGTTCATTGCCCAAAATGTTCTCAGCTGCAGCTTTCATGAGGTAACTCCA
GGGCCCACCTGTTCTCTGGT
>ENSMUSG00000020122|ENSMUST00000125984
GAGTCAGGTTGAAGCTGCCCTGAACACTACAGAGAAGAGAGGCCTTGGTGTCCTGTTGTC
TCCAGAACCCCAATATGTCTTGTGAAGGGCACACAACCCCTCAAAGGGGTGTCACTTCTT
CTGATCACTTTTGTTACTGTTTACTAACTGATCCTATGAATCACTGTGTCTTCTCAGAGG
CCGTGAACCACGTCTGCAAT
```

- The FASTA format's simplicity and flexibility comes with an unfortunate downside: the FASTA format is a loosely defined ad hoc format.

FASTA

- In general, the following rules should be observed:
  1. Sequence lines should not be too long. While a FASTA file that contains the sequence of the entire human chromosome 1 on a single line is a valid FASTA file, most tools that run on such a file would fail.
  2. Some tools may accept data containing alphabets beyond those that they know how to deal with. For example, the standard alphabet for nucleotides would contain ATGC. An extended alphabet may also contain
     1) N: A, T, G, or C
     2) W: A or T
     3) Search the web for “IUPAC” nucleotides to get a list of all such symbols.

FASTA

3. The sequence lines should always at the same width with the exception of the last line. Some tools will fail to operate correctly and may not even warn the users if this condition is not satisfied. The following is technically a valid FASTA but it may cause various problems:

```
>foo
ATGCATGCATGCATGCATGC
ATGCATGCATG
ATGCATGCATGCATGCA
```

It should be reformatted to:

```
>foo
ATGCATGCATGCATGCATGC
ATGCATGCATGATGCATGCA
TGCATGCA
```

FASTA

4. Use upper-case letters. Whereas both lower-case and upper-case letters are allowed by the specification, the different capitalization may carry additional meaning and some tools and methods will operate differently when encoutering upper- or lower-case letters. Some communities (e.g. Ensembl) chose to designate the lower-case letters as all repeats and low complexity regions.

FASTQ

- The FASTQ format extends FASTA by including a numeric quality score to each base in the sequence. The FASTQ format is widely used to store high-throughput sequencing data, which is reported with a per-base quality score indicating the confidence of each base call.

- It is the de facto standard by which all sequencing instruments represent data.

FASTQ

- The FASTQ format looks like:

```
(base) head -n 4 /home/Data/contam.fastq
@DJB775P1:248:D0MDGACXX:7:1202:12362:49613
TGCTTACTCTGCGTTGATACCACTGCTTAGATCGGAAGAGCACACGTCTGAA
+
JJJJJIIJJJJJJHIHHHGHFFFFFFCEEEEEDBD?DDDDDDBDDDABDDCA
(base)
```

- Line1: The description line beginning with @. This contains the record identifier and other information.
- Line2: Sequence data, which can be on one or many lines.
- Line3: The line beginning with + indicates the end of the sequence.
- Line4: Quality data, which can also be on one or many lines, but must be the same length as the sequence. Each numeric base quality is encoded with ASCII characters.

FASTQ

- The FASTQ format is a multi-line format just as the FASTA format is. In the early days of high-throughput sequencing, instruments always produced the entire FASTQ sequence on a single line.

- The FASTQ format suffers from the unexpected flaw that the @ sign is both a FASTQ record separator and a valid value of the quality string. For that reason it is a little more difficult to design a correct FASTQ parsing program.

`bioawk`

- `bioawk` is a program that extends `awk`'s powerful processing of tabular data to processing tasks involving common bioinformatics formats like FASTA/FASTQ, GTF/GFF, BED, SAM, and VCF.

Counting FASTA/FASTQ entries

- Counting FASTA entries:

```
(base) grep -c "^>" /home/Data/egfr_flank.fasta
5
(base)
```

- First approach for counting FASTQ entries:

```
(base) grep -c "^@" /home/Data/untreated1_chr4.fq
208779
(base) wc -l /home/Data/untreated1_chr4.fq
817420 /home/Data/untreated1_chr4.fq
(base) echo "817420/4" | bc
204355
(base)
```

Counting FASTA/FASTQ entries

```
(base) grep "^@" /home/Data/untreated1_chr4.fq | head -n 20
@SRR031729.3941844
@SRR031728.3674563
@SRR031729.8532600
@SRR031729.2779333
@SRR031728.2826481
@SRR031728.2919098
@SRR031729.2873401
@SRR031728.343975
@SRR031729.2496773
@SRR031728.4068187
@SRR031729.4779101
@SRR031729.172565
@SRR031729.4862084
@SRR031729.10091131
@SRR031729.5549475
@SRR031729.10518513
@SRR031729.2324257
@AAB@?A2AA?AAA=???=;A9>:1==3A4=8119=0>1;4A=9,25A11603=;>;411;/0493#########
@SRR031729.7021582
@SRR031729.9697902
(base)
```

Counting FASTA/FASTQ entries

- Second approach for counting FASTQ entries: If you're unsure if some of your FASTQ entries wrap across many lines, a more robust way to count sequences is with `bioawk`

```
(base) /opt/tools/bioawk/bioawk -c fastx 'END{ print NR }' /home/Data/untreated1_chr4.fq
204355
(base)
```

Base qualities in FASTQ

- Each sequence base of a FASTQ entry has a corresponding numeric quality score (Phred score) in the quality line(s).

- Each base quality score is encoded as a single ASCII character, which are represented as integers between 0 and 127. Because not all ASCII characters are printable to screen (e.g. character echoing "`07`" makes a ding noise), qualities are restricted to the printable ASCII characters, ranging from 33 to 126.

- The idea behind the Phred score is to map two digit numbers to single characters so that the length of the quality string stays the same as the length of the sequence.

- ASCII code: https://www.ascii-code.com/

Base qualities in FASTQ

- Converting these ASCII characters to meaningful quality scores can be tricky because there are three different quality schemes: Sanger, Solexa, and Illumina.

Name	ASCII range	Offset	Quality score type	Quality score range
Sanger Illumina (≥1.8)	33-126	33	PHRED	0-93
Solexa Illumina (<1.3)	59-126	64	Solexa	-5-62
Illumina (1.3-1.7)	64-126	64	PHRED	0-62

Base qualities in FASTQ

- Fortunately, the bioinformatics field has finally seemed to settle on the Sanger encoding.

- A Phred score Q is used to compute the probability of a base call being incorrect by the formula

$$P = 10^{-Q/10} \text{ or } Q = -10 \log_{10} P$$

Q	Error	Accuracy
0	1 in 1	0%
10	1 in 10	90%
20	1 in 100	99%
30	1 in 1000	99.9%
40	1 in 10,000	99.99%

Base qualities in FASTQ

- From ASCII characters encoded by Sanger scheme to Phred scores:
  1. Subtract an offset to convert this Sanger quality score to a Phred quality score.

```
(base) head -n 4 /home/Data/contam.fastq
@DJB775P1:248:D0MDGACXX:7:1202:12362:49613
TGCTTACTCTGCGTTGATACCACTGCTTAGATCGGAAGAGCACACGTCTGAA
+
JJJJJIIJJJJJJHIHHHGHFFFFFFCEEEEEDBD?DDDDDDBDDDABDDCA
(base) python -c 'qual="JJJJJIIJJJJJJHIHHHGHFFFFFFCEEEEEDBD?DDDDDDBDDDABDDCA"; print([ord(b)-33 for b in qual]);'
[41, 41, 41, 41, 41, 40, 40, 41, 41, 41, 41, 41, 41, 39, 40, 39, 39, 39, 38, 39, 37, 37, 37, 37, 37, 37, 34, 36, 36, 3
6, 36, 36, 35, 33, 35, 30, 35, 35, 35, 35, 35, 35, 33, 35, 35, 35, 32, 33, 35, 35, 34, 32]
(base)
```

In python, `ord()` converts the ASCII characters to their decimal representations.

Base qualities in FASTQ

2. Apply the formula to convert quality scores to the probability the base is incorrect:

```
(base) python -c 'qual="JJJJJIIJJJJJJHIHHHGHFFFFFFCEEEEEDBD?DDDDDDBDDDABDDCA"; qual=[ord(b)-33 for b in qual];
print([10**(-q/10.0) for q in qual])'
[7.943282347242822e-05, 7.943282347242822e-05, 7.943282347242822e-05, 7.943282347242822e-05, 7.943282347242822e
-05, 0.0001, 0.0001, 7.943282347242822e-05, 7.943282347242822e-05, 7.943282347242822e-05, 7.943282347242822e-05
, 7.943282347242822e-05, 7.943282347242822e-05, 0.00012589254117941674, 0.0001, 0.00012589254117941674, 0.00012
589254117941674, 0.00012589254117941674, 0.00015848931924611142, 0.00012589254117941674, 0.00019952623149688788
, 0.00019952623149688788, 0.00019952623149688788, 0.00019952623149688788, 0.00019952623149688788, 0.00019952623
149688788, 0.00039810717055349735, 0.00025118864315095795, 0.00025118864315095795, 0.00025118864315095795, 0.00
025118864315095795, 0.00025118864315095795, 0.00031622776601683794, 0.00050118723362727 25, 0.00031622776601683794, 0.00031622776601683794, 0.00031622776601683794, 0.00031622776601683794, 0.00
031622776601683794, 0.00031622776601683794, 0.00050118723362727 25, 0.00031622776601683794, 0.00031622776601683794, 0.00031622776601683794, 0.00031622776601683794, 0.00063095734448019 3, 0.00050118723362727 25, 0.00031622776601683794, 0.00031622776601683794, 0.00039810717055349735, 0.00063095734448019 3]
(base)
```

INSTALLING UNIX PROGRAMS

Shell profile

- You may want to apply certain settings automatically when a terminal is opened.

- A file that is processed when a shell is opened is called a shell profile. This file has to have a special name and has to be located in the home directory. All the commands listed in this shell profile will be applied when a new shell is initialized.

Shell profile

- Due to historical and other usage considerations, multiple files (`.bash_profile` and `.bashrc`) may contain shell configurations.

- `.bash_profile`: This file should be set up to automatically load `.bashrc`.

- `.bashrc`: The file is the main shell profile and it should contain all shell related settings. You need to only change the `.bashrc` file.

- Changing a shell profile file will not automatically apply the changed settings to terminals that are already open. New terminals will use the new settings but existing terminal windows need to be instructed to apply the new settings by:

```
source ~/.bashrc
```

Setting up PATH

- Suppose you install the program `fastq-dump` but then as you try to run it you could get this:

```
$ fastq-dump
fastq-dump: command not found
```

- The main concept to remember here is that our computers look at only a few locations when trying to find a program. When bash cannot run a program it is because it cannot see the program. When we install a new program ourselves, we need to tell our computer where to look for it. There are different ways to do this.

Setting up PATH

- Solution 1: Run the program by its fully qualified path.

```
[smkim@ip-172-31-42-146 bin]$ pwd
/opt/tools/sratoolkit.3.0.0-centos_linux64/bin
[smkim@ip-172-31-42-146 bin]$ cd ~
[smkim@ip-172-31-42-146 ~]$ /opt/tools/sratoolkit.3.0.0-centos_linux64/bin/fastq-dump

Usage:
  /opt/tools/sratoolkit.3.0.0-centos_linux64/bin/fastq-dump.3.0.0 [options] <path> [<path>...]
  /opt/tools/sratoolkit.3.0.0-centos_linux64/bin/fastq-dump.3.0.0 [options] <accession>

Use option --help for more information

/opt/tools/sratoolkit.3.0.0-centos_linux64/bin/fastq-dump.3.0.0 : 3.0.0

[smkim@ip-172-31-42-146 ~]$
```

Setting up PATH

- Solution 2: Extend the search PATH to include the installation directory

- If we want to run the program just by typing its name, we need to instruct the computer to look for programs to run in the `/opt/tools/sratoolkit.3.0.0-centos_linux64/bin`

- The shell looks at the variable called `PATH` and searches all locations listed in the `PATH`.

```
echo $PATH
```

Setting up PATH

- To add a new location, type the following code at `.bashrc`:

```
export PATH=$PATH:/opt/tools/sratoolkit.3.0.0-centos_linux64/bin
```

- Run the following command:

```
source ~/.bashrc
```

Installing the SRA toolkit

- The SRA toolkit allows access to the SRA. It is a suite of tools, the most used among them are: `fastq-dump` and `sam-dump`.

- The SRA toolkit is available as executable programs for each platform.

- Visit https://github.com/ncbi/sra-tools

Installing the SRA toolkit (CentOS)

Installing the Entrez Direct (CentOS)

- Entrez Direct consist of a suite of scripts written in the Perl programming language that can be used to connect to the NCBI Entrez data interfaces.

Installing the Entrez Direct (using conda)

```
[yjjeong@ip-172-31-42-146 ~]$ cd ~
[yjjeong@ip-172-31-42-146 ~]$ vi .bashrc
```

```
# .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi

# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=

# User specific aliases and functions

# >>> conda initialize >>>
# !! Contents within this block are managed by 'conda init' !!
__conda_setup="$('/opt/tools/anaconda3/bin/conda' 'shell.bash' 'hook' 2> /dev/null)"
if [ $? -eq 0 ]; then
    eval "$__conda_setup"
else
    if [ -f "/opt/tools/anaconda3/etc/profile.d/conda.sh" ]; then
        . "/opt/tools/anaconda3/etc/profile.d/conda.sh"
    else
        export PATH="/opt/tools/anaconda3/bin:$PATH"
    fi
fi
unset __conda_setup
# <<< conda initialize <<<
```

```
[yjjeong@ip-172-31-42-146 ~]$ source .bashrc
```

```
# >>> conda initialize >>>
# !! Contents within this block are managed by 'conda init' !!
__conda_setup="$('/opt/tools/anaconda3/bin/conda' 'shell.bash' 'hook' 2> /dev/null)"
if [ $? -eq 0 ]; then
    eval "$__conda_setup"
else
    if [ -f "/opt/tools/anaconda3/etc/profile.d/conda.sh" ]; then
        . "/opt/tools/anaconda3/etc/profile.d/conda.sh"
    else
        export PATH="/opt/tools/anaconda3/bin:$PATH"
    fi
fi
unset __conda_setup
# <<< conda initialize <<<
```

Exporting the path (CentOS)

- Add the following line at `.bashrc`

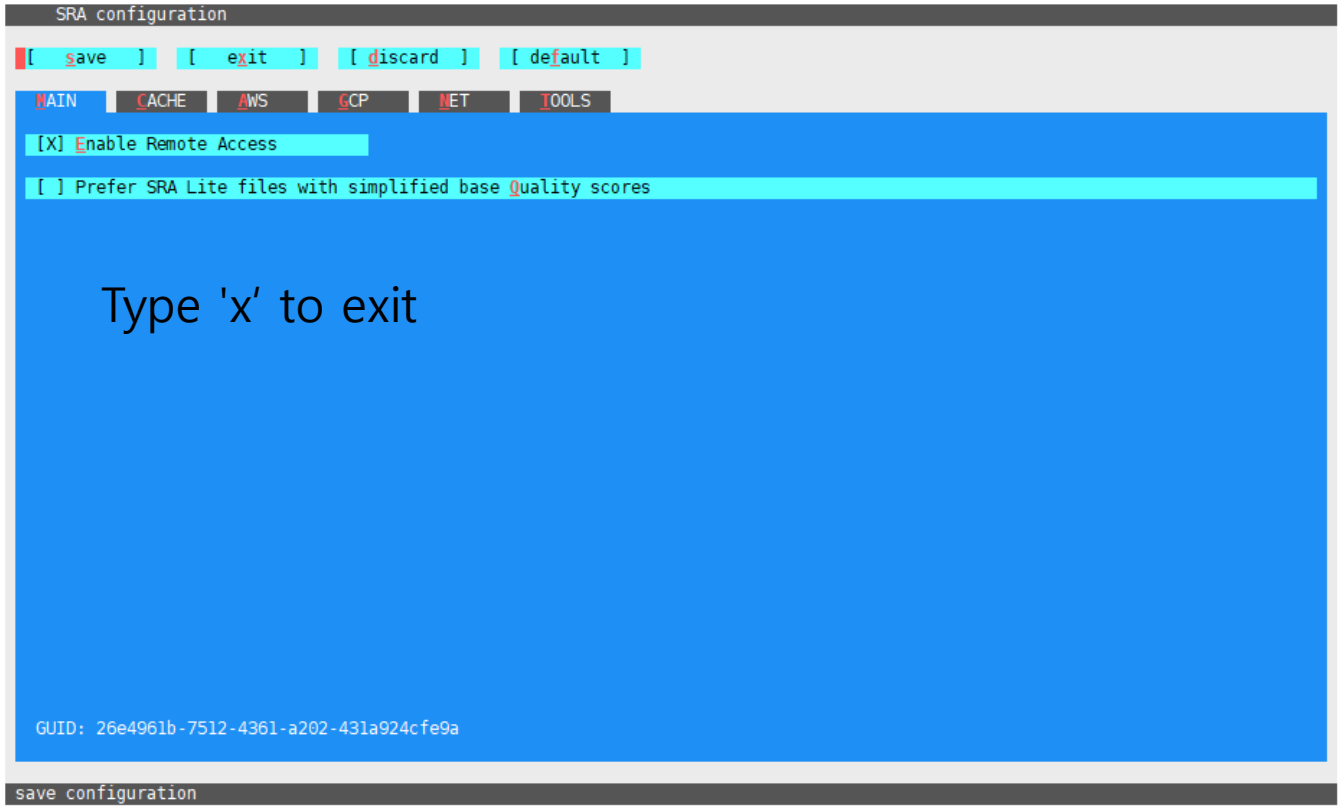
```
export PATH=$PATH:/opt/tools/sratoolkit.3.0.0-centos_linux64/bin
```

- Run the following command

```
source .bashrc
```

Installing the SRA toolkit (CentOS)

```
(base) vdb-config --interactive
```



ACCESSING SRA

Short Read Archive (SRA)

- The SRA is a data storage service provided by NCBI. It stores raw sequencing data and alignment information from high-throughput sequencing platforms.

- https://www.ncbi.nlm.nih.gov/sra

- The GEO (Gene Expression Omnibus) is a public functional genomics data repository. It accepts array- and sequence-based data. Only the gene expression level results are stored in GEO, and the sequence data is deposited into SRA.

- https://www.ncbi.nlm.nih.gov/geo/

Reanalyzing data

- We will learn how to reanalyze the published data from start with your methods. This approach is easier than to retrace the same steps that were published in a scientific paper.

- We want to know how the data has been analyzed not actually to rerun the same commands, but to assess the strengths and weaknesses of the analysis and to identify a possible errors or inconsistencies in the process.

Reanalyzing data

- Genomic surveillance elucidates Ebola virus origin and transmission during the 2014 outbreak

- http://science.sciencemag.org/content/345/6202/1369

- Accessing the raw sequencing data:

> Infectious Diseases grant HHSN272200900049C. Sequence data are available at NCBI (NCBI BioGroup: PRJNA257197). Sharing

SRA naming schemes

- Data in the SRA is organized under a hierarchical structure:
  1. NCBI BioProject: PRJN****** (e.g. PRJNA257197) contains the overall description of a single research initiative. A project will typically related to multiple samples and datasets.
  2. NCBI BioSample: SAMN******** and/or SRS****** (e.g. SAMN03254300) describe biological source material.
  3. SRA Experiment: SRX******** a unique sequencing library for a specific sample
  4. SRA Run: SRR******** is a manifest of data files linked to a given sequencing experiment.

Automating the access to SRA

- Start with the bioproject number `PRJNA257197` for the Ebola virus paper.

```
(base) esearch -db sra -query PRJNA257197
<ENTREZ_DIRECT>
  <Db>sra</Db>
  <WebEnv>MCID_6343c612bccaa869341ae7ac</WebEnv>
  <QueryKey>1</QueryKey>
  <Count>891</Count>
  <Step>1</Step>
</ENTREZ_DIRECT>
(base)
```

- This tells us that there are 891 sequencing runs for this project.

Automating the access to SRA

- Format the output as a “runinfo” format:

```
esearch -db sra -query PRJNA257197 |
efetch -format runinfo > PRJNA257197_runinfo.csv
```

- Cut out the first column and store the run ids:

```
(base) cut -f 1 -d "," PRJNA257197_runinfo.csv | grep "SRR" > PRJNA257197_SRR_ids.txt
(base)
```

Automating the access to SRA

- Our `PRJNA257197_SRR_ids.txt` contains

```
(base) head -n 3 PRJNA257197_SRR_ids.txt
SRR1972948
SRR1972956
SRR1972955
(base)
```

- We can download raw sequencing data using `fastq-dump`:

```
fastq-dump SRR1972917
```

Automating the access to SRA

- By default, the SRA data will concatenate the paired reads because that is actually how the instrument measured them as well. For paired-end reads the data needs to be separated into different files:

```
fastq-dump –split-files SRR1972917
```

Automating the access to SRA

- To automate the process of downloading the raw sequencing data from the list of `PRJNA257197_SRR_ids.txt`, we can use `xargs` command.

```
(base) head -n 2 PRJNA257197_SRR_ids.txt | xargs -n 1 fastq-dump --split-files
Read 573833 spots for SRR1972948
Written 573833 spots for SRR1972948
Read 410593 spots for SRR1972956
Written 410593 spots for SRR1972956
(base) ls
PRJNA257197_runinfo.csv  SRR1972948_1.fastq  SRR1972956_1.fastq
PRJNA257197_SRR_ids.txt  SRR1972948_2.fastq  SRR1972956_2.fastq
(base)
```

xargs

- `xargs` allows us to take input passed to it from standard in, and use this input as arguments to another program. This approach allows us to build commands programmatically from values received through standard in.

- `xargs` passes all arguments received through standard in to the supplied program. This works well for programs like `rm`, `touch`, `mkdir`, and others that take multiple arguments. However, other programs only take a single argument at a time.

- We can set how many arguments are passed to each command call with `xargs`'s `-n` argument.