Exercise URL: https://github.com/djhshih/exercise-rna-seq

Question 1. Write a download script to obtain the paired-end RNA-seq data for the following samples from Sequence Read Archive PRJNA681149:

SUM149PT Parent clone, DMSO treated
SUM149PT C2 clone, DMSO treated

Sol:

Please see the download script for the paired-end RNA-seq data at **fq/get.sh** with an SRA run file ID list at **fq/SRR_Acc_List.txt**.

Question 2. Build a reproducible pipeline to trim the reads with fastp, align the trimmmed reads with STAR, and quantify the expression profiles with rsem.

Sol:

Please see the reproducible pipeline **rna_seq_analysis_pipeline.sh**. Example command to run the pipeline: **task.sh**.

Before running the pipeline, the STAR genome and RSEM genome index can be created by **create_index.sh**.

For the environment configuration of the pipeline, please see **env_config_for_rnaseq.sh** and **rnaseq_env.yml**.

Question3. Describe key steps that need to be added to the pipeline so that it can used in production.

Sol:

To make the pipeline suitable in production, there are some key steps needed to be added:

1. **Check before Running:**
   The pipeline would enable the user to test its reliability and potential problems by dry run. The pipeline would be able to create the output path if the path does not exist.
2. **Logging and Monitoring:**
   Implement logging throughout the pipeline to record important information, the running task progress, errors, and warnings. When encountering error, indicate the location of its occurrence for troubleshooting. The execution and resource usage of the pipeline would also be able to be monitored.
3. **Pipeline Orchestration:**
   Utilize orchestration tools (e.g., Nextflow, Snakemake, cwl, wdl) to manage the pipeline execution.
4. **Reentrancy for Breakpoint Executing:**

Implement strategy to make the pipeline able to continue where it left off if interrupted, without the need to restart from the beginning.

5. **Resource and Time Management:**
   Incorporate resource management strategies to maximize the use of computing resources and prevent exceeding of resource limits. Enable the user to set the computing resources and running time for the task.
6. **Parallelization:**
   Optimize the pipeline for parallel processing to enhance execution efficiency. The pipeline would support the job submission onto multiple nodes.
7. **File Dependency:**
   Implement the update of associated downstream files when a dependency is updated.
8. **File Protection and Deletion:**
   Protect important files produced by modifying permissions of to avoid accidental deletion. Mark the intermediate file as temporary file, which will be deleted after the running.
9. **Documentation:**
   The pipeline should provide explanatory documentation for users, including instructions on how to set up and execute the pipeline, software versions, and document dependencies.
10. **Notification**
    The pipeline will be able to notify users via email or other measures when a task starts or ends or encounters an error interruption.
11. **Report Generation and Data Liberation**
    When the task is finished, the pipeline would be able to generate a result report to summarize and visualize the output results for the user. The data should be able to uploaded to the database or disk customed automatically.
12. **Further Analysis and Statistics on RNA-seq data**
    Add more analysis and statistics steps in the pipeline to make it suitable for clinical application, such as:
    1. Sample correlation analysis
    2. Differential gene expression analysis
    3. GO/KEGG enrichment analysis
    4. Protein interaction analysis
    5. GSEA analysis