#Project 1 R program to crawl, parse and extract all articles published in a specific journal
#Group 1 – Aakash Asli, Shao Yu Chen, Tushar Asnani
#Fall 2018

Readme

- **How to run?**
  - Copy all the R scripts into WD (work directory)
  - First execute util.R
  - After executing util.R, execute loadAllArticles.R

- **loadAllArticles.R**
  - Main Program
  - Step 1 – extract article list page url from the journal's main page.
  - Step 2 – extract total page information (count and url of each page)
  - Step 3 – extract article url list from each page, and store in a data.frame. Call function: loadArticleList()
  - Step 4 – Save all article page (DOI.html) to /HTMLs folder, extract 10 required fields from the article full text page and store in a data.frame. Call function: analysisArticle()
  - Step 5 – save final result(data.frame) into /output/ Genome Biology.txt

- **Util.R**
  - FUNCTION 1: FUNCTION NAME: **loadArticleList**
    FUNCTION: extract DOI and url of article lists in the specified page
    INPUT: [page_url:the url of article list page]
    OUTPUT: [data.frame(DOI,URL:url of article full text)]

  - FUNCTION 2: FUNCTION NAME: **analysisArticle**
    FUNCTION: extract all required field from the specified article full text page
    INPUT: [DOI,URL:url of article full text]
    OUTPUT: [data.frame(DOI, title, author, authorAffiliation, correspondingAuthor, correspondingAuthorEmail, publicationDate, abstract, keywords, fullText)], single row

  - FUNCTION 3: FUNCTION NAME: **extract**
    FUNCTION: extract xmlValue from specified XML tag/node
    INPUT: [parsedHtml, pattern]
    OUTPUT :[string of xmlValue embedded in the tag/node]

  - FUNCTION 4: FUNCTION NAME: **extracAttribute**
    FUNCTION: extract XML attribute value from specified XML tag/node and attribute name
    INPUT: [parsedHtml, pattern, attribute]
    OUTPUT: [string of attribute value] OUTPUT: [string of xmlValue embedded in the tag/node]
  - FUNCTION 5: FUNCTION NAME: **extractAuthors**
    FUNCTION: extract author, corresponding author, corresponding author's email
    INPUT: [parsedHtml]
    OUTPUT: [vector(author, corresponding.author, email)]