[90 min] DO: Build Hierarchical Document Database

Start Assignment

程序代写代做 CS编程辅导

Due Sep 17 by 11:59pm **Points** 100 upload **File Types** r and zip
Available until Sep 20 at 11:59pm

Motivation

Most operating systems use a hierarchical file system for storing documents, images, media, programs, and any other type of data in files. The file system can act as a database and is often used to store large file objects, often in conjunction with a database (such as a relational database). In this assignment, you will experiment with navigating the file system in R and implement a simple query structure for a hierarchical document object data store.

WeChat: cstutorcs

Learning Outcomes

Assignment Project Exam Help

- learn basic R programming
- use the file system as a document store
- appreciate the use of lock files to manage concurrency

Email: tutorcs@163.com

Format

QQ: 749389476

May be done in pairs (groups of two) or individually. If done in pairs, both team members must make an individual (and unique) submission and clearly indicate the name of the collaborator in both the submitted file and in a submission comment. Collaborators cannot submit the same code.

https://tutorcs.com

Due Date

This is one of the few assignments that does not have a due date at the end of the module week but we strongly urge you to complete the assignment after you go through the first module. It is extremely helpful in learning R. The reason why there's no usual end-of-module due date is simple: students often sign up for this course after the term starts and thus need time to complete this assignment. There are no submissions allowed past the due date, so no "late submissions".

Material Needed

- R and R Studio OR **rstudio.cloud** **(http://rstudio.cloud)**

Note that if you choose to use **rstudio.cloud** **[illegible])** you will likely need to create a paid account to gain sufficient usage time. An educational account is available. We recommen[illegible] and R Studio locally.

Prerequisites

Prior to working on this assignment, review these lessons and refer to them during the assignment:

- **6.104 Quick Guide to R for Programmers** **(http://artificium.us/lessons/06.r/l-6-104-r4progs/l-6-104.html)**
- **6.202 Working with R Projects** **(http://artificium.us/lessons/06.r/l-6-202-r-projects/l-6-202.html)**
- **6.109 R Scripts and Programs** **(http://artificium.us/lessons/06.r/l-6-109-r-programs/l-6-109.html)**
- **6.190 Console Output in R** **(http://artificium.us/lessons/06.r/l-6-190-console-output-r/l-6-190.html)**
- **6.402 Navigating the File System in R** **(http://artificium.us/lessons/06.r/l-6-402-filesystem-from-r/l-6-402.html)**
- **6.121 Writing Functions in R** **(http://artificium.us/lessons/06.r/l-6-121-funcs-in-r/l-6-121.html)**
- **6.112 Basics of Text & String Processing in R** **(http://artificium.us/lessons/06.r/l-6-112-text-proc/l-6-112.html)**

Tasks

The tasks below assume that you have installed R and R Studio or created an account on **rstudio.cloud** **(http://rstudio.cloud)** . The tasks below guide you through the process of creating a document store that uses folders as a means to organize data. The "records" that are stored are images. Images can be tagged. Each folder represents a tag. For example, the image file CampusAtNight.jpg might have associated tags "#Northeastern" and "#ISEC", so the file is then stored (twice) in the folders "Northeastern" and "ISEC". Of course, this isn't super efficient and could be improved with symbolic links, but for now that is the implementation.

Add files for you to do the testing as you see fit.

Before launching into the tasks below, watch the commentary and explanation:

程序代写代做 CS编程辅导
Tutor CS
WeChat: cstutorcs
Assignment Project Exam Help
Email: tutorcs@163.com
QQ: 749389476
https://tutorcs.com

程序代写代做 CS编程辅导

1. (*1 min / 0 pts*) Launch R Studio and create a[illegible]CS5200.BuildDocDB.*LastName*" where *LastName* is your last name.
2. (*1 min / 0 pts*) In the R Project, create an R program (script) titled "*ObjDB-LastName.R*" where *LastName* is your last name.
3. (*5 min / 5 pts*) R programs run as a script starting with the first line. Adopting the mechanism from C/C++, make the first line of code of the R program a call to the function `main()` and the second line a call to `quit()` which works like the function *exit()* in C/C++. Then write a function called `main()` before the call to `main()` that will eventually call all other functions we will build below. All of your "testing code" will eventually be in `main()`. We will not use any kind of unit testing packages. All code must be in the function `main()` or some other function. Only global variables can be declared outside of `main()`. The code fragment below shows this approach.

```
globalVar <- 0

main <- function()
{
  # all program code starts here
  print ("Hello, World")
}

##########################################################

main()
```

4. (*5 min / 5 pts*) Add a global variable before `main()` called *rootDir* that has the value "docDB".

WeChat: cstutorcs
Assignment Project Exam Help
Email: tutorcs@163.com
QQ: 749389476
https://tutorcs.com

5. (*20 min / 5 pts*) Write a function called `configDB(root, path)` that sets up all folders and database related structure. For now that is just the folder in which all tag folders will be stored, *e.g.*, assuming that the value "docDB" is passed for *root*, the function it creates the folder "docDB" in the project folder if the *path* argument is empty (*i.e.*, "") or under the provided path.
6. (*10 min / 5 pts*) Write a function called `genObjPath(root, tag)` that returns the correctly generated path to a tag folder, *e.g.*, if *tag* is #ISEC it would return "docDB/ISEC". Note the stripped # in the path.
7. (*10 min / 15 pts*) Write a function called `getTa`[illegible] [illegible]eturns a vector of tags in the file name, *e.g.*, if the *fileName* argument has the value "CampusAtNight.jpg #Northeastern #IS[illegible] MacOS or Linux) or "CampusAtNight #Northeastern #ISEC.jpg" (on Windows where the extension of the file must be at the [illegible] [illegible]rn the vector *("#ISEC", "#Northeastern")*. Note that on Windows files names, the extension at the end of part of the file nar[illegible] a tag, so the tags for the Windows file "CampusAtNight #Northeastern #ISEC**.jpg**" are "#Northeastern" and "ISEC"; [illegible] tension to indicate an image file and part of the file name: "CampusAtNight**.jpg**".
8. (*10 min / 10 pts*) Write a function called `getFileName(fileName)` that returns file name, *e.g.*, if the *fileName* argument has the value "CampusAtNight.jpg #Northeastern #ISEC" or "CampusAtNight #Northeastern #ISEC.jpg" it should return the string "CampusAtNight.jpg".
9. (*10 min / 20 pts*) Write a function called `storeObjs(folder, root)` that copies all files in the specified in the *folder* argument to their correct folders underneath the *root* folder. Create folders for the tags as needed. The file must be stored in the "tag folders" without the tags, *e.g.*, the image file "CampusAtNight.jpg #Northeastern #ISEC" or "CampusAtNight #Northeastern #ISEC.jpg" should be stored under the name "CampusAtNight.jpg" in the folders "docDB/ISEC" and "docDB/Northeastern" assuming that *root* has the value "docDB". Leverage all of the functions developed previously when building `storeObjs(folder, tags)`.
10. (*10 min / 5 pts*) Modify the function `storeObjs(folder, root)` created above so that it takes a third argument *verbose* that is a boolean. If the argument is true, modify the code for the function so that it prints a message for every file that is copied. The message should have the form: "Copying CampusAtNight.jpg to ISEC, Northeastern". In general, it should print the name of the file being copied and the tags separated by commas
11. (*10 min / 10 pts*) Write a function called `clearDB(root)` that removes all folders and files in the folder specified by *root* but not the folder for *root* itself. This function is used to "reinitialize" the database to a "blank" state.
12. (*10 min / 10 pts*) Add code to `main()` to demonstrate that your functions are working.
13. (*10 pts*) Verify that your code is properly structured and documented and follows generally accepted programming practices. Write as much documentation as you need to communicate to other what you have done and to ensure that others can understand your thought process, your code, and any assumptions or exceptions. Use function headers to explain the signature of the function.

程序代写代做 CS编程辅导
WeChat: cstutorcs
Assignment Project Exam Help
Email: tutorcs@163.com
QQ: 749389476
https://tutorcs.com

Hints

- post questions to the Teams channel
- there are other ways we could have architected this "hierarchical file database" -- but this is the way we chose to do this and it is sufficient to learn some R and to see how file systems can act as databases
- don't use any unit testing packages
- you may assume that periods are not allowe[illegible] .bar or #pic.jpg would not be legal tags; but keep in mind that on Windows, for the file "foo #bar.jpg", the ".jpg" is NOT part o[illegible]sents the file extension
- you should account for file names that have t[illegible]*.g.*, .jpg or .mp3 or .tiff) either at the end of the file name stem and before the tags (as would be the case on Unix or MacO[illegible]nd (as would be the case for MS-DOS or Windows)
- you must accommodate any extension, inclu[illegible]night not think of (the final part of the file name after the last dot/period is the extension)
- do not put your *.R* source files into the "docDB" directory -- source files are not part of the database (obviously)

Submission

Submit the *ObjDB-LastName.R* program containing your code. Programs that do not run or throw an error during execution will not receive any credit.

CS5200.S23.DocDB-in-R

程序代写代做 CS编程辅导
WeChat: cstutorcs
Assignment Project Exam Help
Email: tutorcs@163.com
QQ: 749389476
https://tutorcs.com

Criteria	Ratings					Pts
Code implements requirements and contains required functions.	**70 pts** **Flawless, no defects**	**60 pts** **Minor defects but otherwise code works**	**30 pts** **Significant defects but code works partially**	**0 pts** **Major defects, or code does not run**		70 pts
Test cases provided for all working scenarios, plus abnormal use cases including missing files and directories	**1[illegible]** **F[illegible]** **f[illegible]** **c[illegible]**	**[illegible]0 pts** **Good test cases but not perfect**	**7 pts** **Reasonably good test cases**	**4 pts** **Some test cases but critical ones are missing**	**0 pts** **No Marks**	15 pts
Code well document, functions have headers, program files contain headers with author information	**10 pts** **Full Documentation**	**5 pts** **Acceptable**	**0 pts** **Little to no documentation or documentation is not helpful**			10 pts
Files named as required	**5 pts** **Fully Meets Requirements**	**3 pts** **Some Minor Mistakes**	**0 pts** **Does not meet requirements or has significant mistakes**			5 pts
						Total Points: 100

程序代写代做 CS编程辅导

WeChat: cstutorcs

Assignment Project Exam Help

Email: tutorcs@163.com

QQ: 749389476

https://tutorcs.com