#1

```
function listMyFiles(folder) {
   const files = folder.getFiles();
   while (files.hasNext()){
     const file = files.next();
     Logger.log(file.getName());
   }
 }
  function filesinFolder(){
   const id = '1VqomA-OF_Gd8j20fZB1f5YiddQdIyuzL';
   const folder = DriveApp.getFolderById(id);
   listMyFiles(folder);
  }
```

#2

```
function makeIt(){
   const html = '<h1>Hello World</h1>';
   const id = '1VqomA-OF_Gd8j20fZB1f5YiddQdIyuzL';
   const folder = DriveApp.getFolderById(id);
   const newFolder = folder.createFolder('New Folder');
   newFolder.createFile('New HTML File',html,MimeType.HTML);
 }
```

#3

```
function driveDetails(){
   const drive1 = DriveApp.getStorageLimit();
   const drive2 = DriveApp.getStorageUsed();
   Logger.log(drive2/drive1 *100);
 }
```

#4

```
function movetoTrashFiles(){
   const files = DriveApp.getFilesByName('New HTML File');
   while (files.hasNext()){
     const file = files.next();
     const today = new Date();
     const fileInfo = {};
     const fileDate = file.getLastUpdated();
     fileInfo.active = today - fileDate;
     fileInfo.nameFile = file.getName();
     fileInfo.fileType = file.getMimeType();
     const hour1 = 1000 * 60 * 60;
     Logger.log(hour1);
```

```
      if(fileInfo.active < hour1){
      file.setTrashed(true);
    }
    Logger.log(fileInfo);
  }
}
 function getFromTrash(){
  const files = DriveApp.getTrashedFiles();
  while (files.hasNext()){
    const file = files.next();
    const today = new Date();
    const fileInfo = {};
    const fileDate = file.getLastUpdated();
    fileInfo.active = today - fileDate;
    const hour1 = 1000 * 60 * 60;
    Logger.log(hour1);
    if(fileInfo.active < hour1){
      file.setTrashed(false);
    }
  }

 }
```

#5

```
function myFiles(){
  const id = '1j0DLOdUARddc1PueTzelZh85nllT4-dg';
  const folder = DriveApp.getFolderById(id);
  const files = folder.getFiles();
  while (files.hasNext()){
    const file = files.next();
    const editors = file.getEditors();
    Logger.log('*****'+file.getName()+'*****');
    editors.forEach((editor)=>{
      outputDriveUser(editor);
      file.removeEditor(editor.getEmail());
    })
    file.setSharing(DriveApp.Access.ANYONE, DriveApp.Permission.VIEW);//Access
Permissions - Permission type
    file.addViewer('education2learning@gmail.com'); //Permission for file
  }
}
```

```
function myPermissions(){
  const id = '1j0DLOdUARddc1PueTzelZh85nllT4-dg';
  const folder = DriveApp.getFolderById(id);
  const editors = folder.getEditors();
   editors.forEach((editor)=>{
    outputDriveUser(editor)
  })
  const viewers = folder.getViewers();
  viewers.forEach((person)=>{
    outputDriveUser(person);
  })
  const owner = folder.getOwner();
  outputDriveUser(owner);
}

function outputDriveUser(user){
    Logger.log(user.getEmail());
    Logger.log(user.getName());
}
```

#6

```
function createaPDF(){
  const id = '1j0DLOdUARddc1PueTzelZh85nllT4-dg';
  const folderSource = DriveApp.getFolderById(id);
  const folderDes = DriveApp.createFolder('PDF');
  const files = folderSource.getFiles();
  while (files.hasNext()){
    const file = files.next();
    const blobFile = file.getAs('application/pdf');
    blobFile.setName('PDF '+file.getName()+'.pdf');
    const newFile = folderDes.createFile(blobFile);
    Logger.log(newFile.getUrl());
  }
}
```

#7

```
function getDetails(){
  const id = '1hGsVI-FJ3fcsNsNE7lFSMvHwV04k352m';
  const folder = DriveApp.getFolderById(id);
  const files = folder.getFiles();
  const ss = SpreadsheetApp.create(folder.getName());
```

```
  const sheet = ss.insertSheet('File Info');
  const ssid = ss.getId();
  ss.appendRow(['URL','Size','Download','id']);
  while (files.hasNext()){
    const file = files.next();
    const fileInfo = [];
    fileInfo.push(file.getUrl());
    fileInfo.push(file.getSize());
    fileInfo.push(file.getDownloadUrl());
    fileInfo.push(file.getId());
    Logger.log(fileInfo);
    ss.appendRow(fileInfo);
  }
}
```

#8

```
function moveFiles(){
  const id = '1hGsVI-FJ3fcsNsNE7lFSMvHwV04k352m';
  const srcFolder = DriveApp.getFolderById(id);
  const endFolder = DriveApp.createFolder('moved');
  const files = srcFolder.getFiles();
  while (files.hasNext()){
    const file = files.next();
    endFolder.addFile(file);
  }
}
 function moveCreate(){
  const ss = SpreadsheetApp.create('New File');
  const folders = DriveApp.getFoldersByName('moved');
  const ssid = ss.getId();
  const file = DriveApp.getFileById(ssid);
  if(folders.hasNext()){
    let folder = folders.next();
    folder.addFile(file);
  }
  Logger.log(file);
}
```

#9

```
function searcher(val){
  const id = '1_K6euAToeLOTJqfM1uNtzickbtJi8VIx';
  const folder = DriveApp.getFolderById(id);
  const result = folder.searchFiles('title contains "'+val+'"');
```

```
  //const result = DriveApp.searchFiles('title contains "'+val+'"');
  const res = [];
  while (result.hasNext()){
    const file = result.next();
    const results = {};
    results.id = file.getId();
    results.url = file.getName();
    results.size = file.getSize();
    results.owner = file.getOwner();
    results.type = file.getMimeType();
    res.push(results);
  }
  return res;
}

function startSearch(){
  const searchTerm = 'new';
  const found = searcher(searchTerm);
  Logger.log(found);
}
```

https://developers.google.com/drive/api/v2/ref-search-terms

Operator	Usage
contains	The content of one string is present in the other.
=	The content of a string or boolean is equal to the other.
!=	The content of a string or boolean is not equal to the other.
<	A value is less than another.
<=	A value is less than or equal to another.
>	A value is greater than another.
>=	A value is greater than or equal to another.
in	An element is contained within a collection.
and	Return items that match both queries.
or	Return items that match either query.

not Negates a search query.

has A collection contains an element matching the parameters.

Term	Valid operators	Usage
title	contains1, =, !=	Name of the file. Surround with single quotes '. Escape single quotes in queries with \', e.g., 'Valentine\'s Day'.
fullText	contains2 3	Full text of the file including title, description, content, and indexable text. Whether the 'title', 'description', or 'indexableText' properties, or the content of the file matches. Surround with single quotes '. Escape single quotes in queries with \', e.g., 'Valentine\'s Day'.
mimeType	contains, =, !=	MIME type of the file. Surround with single quotes '. Escape single quotes in queries with \', e.g., 'Valentine\'s Day'.
modifiedDate	<=, <, =, !=, >, >=	Date of the last modification of the file. RFC 3339 format, default timezone is UTC, e.g., 2012-06-04T12:00:00-08:00. Fields of type date are not currently comparable to each other, only to constant dates.
lastViewedByMeDate	<=, <, =, !=, >, >=	Date that the user last viewed a file. RFC 3339 format, default timezone is UTC, e.g., 2012-06-04T12:00:00-08:00. Fields of type date are not currently comparable to each other, only to constant dates.
trashed	=, !=	Whether the file is in the trash or not. Can be either true or false.
starred	=, !=	Whether the file is starred or not. Can be either true or false.
parents	in	Whether the parents collection contains the specified ID.
owners4	in	Users who own the file.
writers4	in	Users or groups who have permission to modify the file. See Permissions resource reference.
readers4	in	Users or groups who have permission to read the file. See Permissions resource reference.
sharedWithMe	=, !=	Files that are in the user's "Shared with me" collection. All file users are in the file's access control list (ACL). Can be either true or false.
properties	has	Public custom file properties.

```
visibility    =, '!='       The visibility level of the file. Valid values are
anyoneCanFind, anyoneWithLink, domainCanFind, domainWithLink, and limited. Surround
with single quotes '. Escape single quotes in queries with \', e.g., 'Valentine\'s
Day'.
```

#10

```
function onOpen(){
   const ui = SpreadsheetApp.getUi();
   ui.createMenu('adv')
   .addItem('search','findDrive')
   .addToUi();
 }
  function findDrive(){
   const ui = SpreadsheetApp.getUi();
   const res = ui.prompt('Search','Search for what',ui.ButtonSet.OK_CANCEL);
   if (res.getSelectedButton() == ui.Button.OK){
     const searchTerm = res.getResponseText();
     Logger.log(searchTerm);
     const results = searchDrive(searchTerm);
     if(results){
       results.forEach((data)=>{
         Logger.log(data);
         addtoSheet(data);
       })
     }
   }
 }
  function addtoSheet(data){
     const ss = SpreadsheetApp.getActive().getSheetByName('results');
     ss.appendRow(data);
 }
  function searchDrive(val){
   const results = DriveApp.searchFiles('title contains "'+val+'"');
   const ret = [];
   while (results.hasNext()){
     const file = results.next();
```

```
    const temp = [file.getName(),file.getOwner(),file.getUrl()];
    ret.push(temp);
  }
  return ret;
}
```