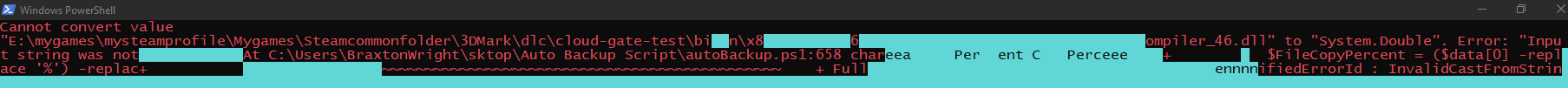
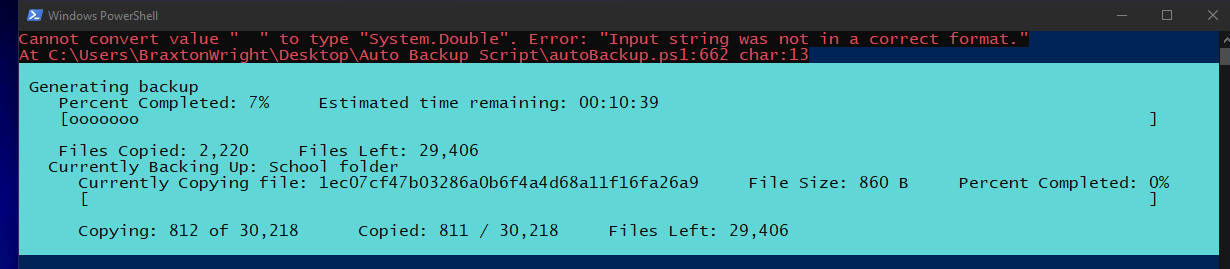
1. Need to see about implementing Start-Threadjob for the counting of the files before the backup progress to speed up the counting of the files. In addition, after you start the jobs, you need to make it so that it automatically starts backing up to make it as fast as possible and the jobs will return the file count after they are done counting, this is because of files like inside unity. One project along contains around 15,000 files. File explore both starts to copy the files while it is counting them and it updates the files remaining portion of the backup.

Also, see about debugging the process and see what is causing it to take so long. This is because when you try to do the job “Unity Projects” it will freeze at the counting of the files and I do not know at what step it is causing the issue. If it is the counting of the files from the source/destination or comparing the objects. Or see if this issue is fixed because of the change of implementation of counting the files.

1. See about finding some logic to better calculate the time remaining for the backup/restore process. It currently does not take into account the size or copy rate of the files, only the number of files copied.
2. See if you can optimize the code to run faster? See if it needs optimized, test by timing how long it takes to copy a set of files using file explorer, a bare bone copy of the robocopy script, and your current iteration of the script.
3. Add a job summary once the backup/restore process has completed. Such as how long it took to complete, how many files were processed, ect. Like how the end of a robocopy, there is that summary table for the job.
4. There is some issue with copying files, it will get a string when it is getting the file size for the function Get-RobocopyProgress. At one time, I got this error in running the script, saying it can’t convert a " " to type "System.Double". However, I can’t seem to replicate this issue I received?
5. Need to see about adding some logic to allow users to continue with the backup/restore if a source directory does not exist. Such as a game does not have any saves, the Game/OS will not yet have made the required file structure for said file saves. In addition, if a new set of destination paths have been defined but have not been backed up to, then they would not yet exist.
6. Need to have it so if a drive is not detected, it will not return the “Path Errors detected” error list. (Cannot seem to replicate and the logic implies it should be impossible because we are using an elseif statement).
7. Need to modify the logic for the first write-progress parameters because when the first start backing up the files, you get a situation where the “Files Copied“ field will say -1 / # as shown below as I was editing a video of the process running.  
   
8. See about adding an option to allow the user to choose if they want a progress bar to be shown or not, because it will result in the process being faster I think.