**Lesson 15 – More Bash programming concepts**

**Issues:**

1. **How to debug a bash script?**
   1. Run the bash script in "debugging mode" using the –x option.  
      Change the first line of the file to #!/bin/bash –x

Displays each line after command substitution but before it is executed.

* 1. Use "verbose mode" by specifying the –v option. (#!/bin/bash –v)  
     Displays each line of the script before command substitution.
  2. Turn on and off debugging mode for sections of the script.  
     set –x # activate debugging from here  
     cmd(s)  
     set +x # stop debugging from here

1. **Windows vs. Unix file names**
   1. Cygwin tools require Unix file names, but Windows programs require Windows file names.
   2. Use "cygpath –w "$filename" to convert UNIX file names to their Windows equivalent.
   3. Use "cygpath –u "$filename" to convert Windows file names to their UNIX equivalent.
2. **A summary of brackets on the command line and in bash scripts:**
   1. ( cmds ) indicates a subshell (i.e., a list of commands that are executed in a separate sub-process); any redirection, assignment, etc. performed inside the parentheses has no effect outside the parentheses.
   2. { cmds; } executes a list of commands in the current shell context.
   3. $( cmds ) is a command substitution. The output of the commands in(cmds) becomes a part of the command line.
   4. $((…)) surround an arithmetic instruction, which expands to the integer value of the expression (stored as a string).
   5. [[ expression ]] surround conditional expressions.
   6. [ expression ] is an alternate form of conditional expressions with more quirks. "Old school" style syntax. (Avoid this syntax.)
3. **Expressions are tricky!** (always put a space before and after variables and operators)
   1. $foo == a+ true if $foo contains a series of 1 or more a's (regular expression pattern matching)
   2. $foo == "a+" true if $foo contains the substring "a+"
   3. $foo =~ a+b\* **extended** regular expression pattern matching
   4. $foo =~ "ab" **extended** regular expression pattern matching

Experiment with a command line like:  
echo $(if [[ "aaaabbb" =~ "aa" ]]; then echo "true"; else echo "false"; fi)

1. **Remember the difference in quotes:**
   1. 'exactly this' **vs.**  "substitutions happens inside double quotes, like $a"

**Problem:** Convert every "IBM form viewer" file (\*.xfdl) to a PDF file (because the IBM form viewer is being removed from all DOD computers).

**Algorithm overview:** There is no tool to convert "IBM Forms viewer" documents to PDF. However, the "Form viewer" program does have functionality to print. And there is a PDF printer driver on your computer that can take a "print document" and "print" it to a PDF file. Therefore, we want an efficient way to "print" every \*.xfdl file on your computer.

**Algorithm:**

1. Before you run your bash script: Set your default printer to "Adobe PDF"
2. Find all of the files in your file hierarchy that have the extension ".xfdl"
3. For each file:
   1. Convert the filename to Windows format (use cygpath)
   2. Execute the "IBM Forms viewer" on the file with the /p option, which sends the document to the printer. On Dr. Brown's computer, the path to the form viewer is  
        
      '/cygdrive/c/Program Files (x86)/IBM/Forms Viewer/4.0/masqform.exe'

Note: The PDF files are "printed" to the default home folder. On Dr. Brown's computer this is "C:/Users/Wayne.Brown/My Documents". If you go to the "Devices and Printers" control panel, right-click on the "Adobe PDF" printer icon and select "Printing Preferences", you can change the default output folder to "folder of the source file".

**A possible solution:**

#!/bin/bash

echo 'This script will print IBM Form viewer files, or convert them'

echo 'to PDF one at a time by printing to the PDF print driver.'

echo ''

echo 'Before you use this script, you need to change the default'

echo 'printer to PDF. Then, for each .xfdl file it finds, hit'

echo 'the save button.'

echo ''

echo 'Do you want to proceed? (y/n)'

read doIt

if [[ $doIt == 'y' || $doIt == 'Y' ]]; then

IFS=$'\n' # The seperator between file names is \n

startingFolder='/cygdrive/c/Users/Wayne.Brown/Documents/USAFA/AIC/Form 94s'

formViewer='/cygdrive/c/Program Files (x86)/IBM/Forms Viewer/4.0/masqform.exe'

for n in $(find $startingFolder -name "\*.xfdl"); do

echo "$n"

fileName=$(cygpath -w "$n")

echo "Converting $fileName to PDF"

"$formViewer" /p "$fileName"

done

fi