

```
1  #! /bin/sh
2
3  #declarations
4  ZIP_ARCHIVES="Complaints/Zip_Archives"
5  FILE_ARCHIVES="Complaints/File_Archives"
6  PROCESSING="Complaints/Processing"
7  MAIN_DIRECTORY="Complaints"
8  MASTER_FILE="Complaints/Complaints_Master.csv"
9
10 #ensure in home directory
11 cd ~
12
13 setup(){
14
15     #make or remake the main folder structure
16     if [[ -d $MAIN_DIRECTORY ]];then
17         rm -r $MAIN_DIRECTORY
18         mkdir -p "$FILE_ARCHIVES"
19         mkdir -p "$ZIP_ARCHIVES"
20         mkdir -p "$PROCESSING"
21     else
22         mkdir -p "$FILE_ARCHIVES"
23         mkdir -p "$ZIP_ARCHIVES"
24         mkdir -p "$PROCESSING"
25     fi
26
27     #setup master file with header
28     if ! [[ -f $MASTER_FILE ]];then
29         echo "Complaint ID,Date Received,Company,Product,Issue" >
... $MASTER_FILE
30
31     fi
32
33 }
34
35 option_1(){
36
37     #tar -xzf aggregate_complaints_001.tar.gz -C $FILE_ARCHIVES #THIS IS FOR
... TESTING ONLY
38
39     #create a list of all zip files to process
40     active_zip_file_list=("aggregate_complaints"*)
41
42     #unpack each zip file then move the zip file to the zip archive
43     for i in ${active_zip_file_list[*]};
44     do
45         tar -xzf $i -C "$PROCESSING"
46         cp $i "$ZIP_ARCHIVES/$i"
47     done
```

```
48
49     #notify the user of a wait
50     echo "Please wait, processing "${#active_zip_file_list[*]}" zipped
... complaint file(s)"
51
52     #populate arrays of file names in processing and in file archives
... WARNING: THESE ARE FULL PATHS
53     active_file_list=("$PROCESSING"/*)
54     archive_file_list=("$FILE_ARCHIVES"/*)
55
56     #ensure that we don't process a file that has previously been processed
57     for i in ${active_file_list[*]};
58     do
59         for j in ${archive_file_list[*]};
60         do
61             i=$(echo $i | cut -d "/" -f 3))      #extracting just the
... file name
62             j=$(echo $j | cut -d "/" -f 3))      #extracting just the
... file name
63             if [[ $i == $j ]];then
64                 rm $PROCESSING/$i                #keep in mind that
... $i is now just the file name
65                 echo $i" deleted"
66                 fi
67             done
68         done
69
70
71     #process all files in processing, move them to file archives as they are
... processed
72     #issues: skipping first line, last line maybe
73     active_file_list=("$PROCESSING"/*)
74     for i in ${active_file_list[*]};
75     do
76         #i=$(echo $i | cut -d "/" -f 3))      #extracting just the file name
77         #echo $i
78
79         while IFS= read -r line
80         do
81             linelen=${#line}
82             if [[ linelen -eq 2 ]];then          #this should work to catch the
... last line as well
83                 continue
84             fi
85             line=$(echo $line | tr " " "_")
86             mya=$(echo $line | cut -d ":" -f 1- --output-delimiter=" ")
87
88             #get complaint id
89             cid=${mya[-1]}
```

```

90         cid=$(echo $cid | cut -d "\" -f2)
91
92         #get date received
93         date_received=${mya[1]}
94         date_received=$(echo $date_received | cut -d "\" -f2)
95
96         #get company WARNING company is messed up
97         company=${mya[ -11]}
98         company=$(echo $company | tr "," "_")
99         company=$(echo $company | cut -d "\" -f2)
100
101         #get product
102         product=${mya[2]}
103         product=$(echo $product | tr "," "_")
104         product=$(echo $product | cut -d "\" -f2)
105
106
107         #get issue
108         issue=${mya[4]}
109         issue=$(echo $issue | tr "," "_")
110         issue=$(echo $issue | cut -d "\" -f2)
111
112         #write the data to the file
113         output="$cid,$date_received,$company,$product,$issue"
114         echo $output >> $MASTER_FILE
115
116         #read < /dev/tty TESTING PURPOSES ONLY
117         done < $i
118
119         #move the processed file
120         i=$(echo $i | cut -d "/" -f 3))
121         mv "$PROCESSING/$i" "$FILE_ARCHIVES/$i"
122
123     done
124 }
125
126
127 option_2(){
128     #cleanup
129
130     #load an array of all file records
131     while IFS= read -r line
132     do
133         #echo $line
134         if [[ $line == "Complaint ID,Date
... Received,Company,Product,Issue" ]];then      #skip the header
135             continue
136         fi
137

```

```
138         cid_list+=($(echo $line | cut -f 1-))
139
140         #read < /dev/tty TESTING PURPOSES ONLY
141     done < $MASTER_FILE
142
143     current_count=${#cid_list[*]}
144     echo "Number of current records: $current_count"
145
146     cid_list=$(
147         for i in ${cid_list[*]};
148         do
149             echo $i
150         done | sort -u)
151
152     post_count=${#cid_list[*]}
153     echo "Number of records after removing duplicates: $post_count"
154     echo "Duplicate records removed: $(( $current_count - $post_count
... ))"
155
156     #now write that new list to the csv file (but use a duplicate for
... testing)
157     echo "Complaint ID,Date Received,Company,Product,Issue" >
... "$MASTER_FILE"
158     for i in ${cid_list[*]};
159     do
160         echo $i >> "$MASTER_FILE"
161     done
162
163     read -p "Press Enter to continue: "
164
165 }
166
167 option_3(){
168     #reporting
169     clear
170     #initialize list and load all records into csv_loaded list
171     while IFS= read -r line
172     do
173         #echo $line
174         if [[ $line == "Complaint ID,Date
... Received,Company,Product,Issue" ]];then      #skip the header
175             continue
176         fi
177
178         csv_loaded+=($(echo $line | cut -f 1-))
179
180         #read < /dev/tty TESTING PURPOSES ONLY
181     done < $MASTER_FILE
182
```

```
183 #load products into product_list
184 for i in ${csv_loaded[*]};
185 do
186     product=$(echo $i | cut -d "," -f4)
187     product_list+=($product)
188 done
189
190 #solicit products and show reporting
191 while true;
192 do
193     matches=1
194     issue_list=()
195     #product_list=()
196     company_list=()
197     clear
198     echo "AVAILABLE PRODUCTS"
199     echo "-----"
200
201     #remove duplicate products
202     product_list=$(
203         for i in ${product_list[*]};
204         do
205             echo $i
206         done | sort -u)
207
208     #list the products
209     for i in ${!product_list[*]};
210     do
211         i=$(( $i + 1 ))
212         product=$(echo ${product_list[$i-1]} | tr "_" " ")
213         product=$(echo $product | sed 's/ / /g')
214         if [[ $i -lt 10 ]];then
215             echo " $i $product"
216         else
217             echo "$i $product"
218         fi
219     done
220
221     #get the user product number choice
222     echo
223     read -p "Enter the product number (zero to exit): " choice
224     my_count=${#product_list[*]}
225     #error trapping for correct input
226     #echo $choice
227     if [[ $choice -eq 0 ]];then
228         break
229     fi
230
231     #get the proper product from the product list, based on the user's
```

```
231... numerical input
232     choice_text=${product_list[$choice -1]}
233
234     #get the matching issues and companies
235     for i in ${csv_loaded[*]};
236     do
237         product=$(echo $i | cut -d "," -f4)
238         if [[ $product == $choice_text* ]];then
239             issue=$(echo $i | cut -d "," -f5)
240             issue_list+=($issue)
241             company=$(echo $i | cut -d "," -f3)
242             company_list+=($company)
243         fi
244     done
245
246     #get a count of matching records (issue list length with dups)
247     matches=${#issue_list[*]}
248
249     #remove duplicates from the issue list
250     issue_list=(
251     for i in ${issue_list[*]};
252     do
253         echo $i
254     done | sort -u)
255
256     #remove duplicates from the company list
257     company_list=(
258     for i in ${company_list[*]};
259     do
260         echo $i
261     done | sort -u)
262
263     #show the report elements
264     clear
265     product=$(echo $choice_text | tr [:lower:] [:upper:])
266     choice_text=$(echo $choice_text | tr [:lower:] [:upper:])
267     choice_text=$(echo $choice_text | tr "_" " ")
268     choice_text=$(echo $choice_text | sed 's/ / /g')
269     echo "PRODUCT: $choice_text"
270     echo "Number of companies involved: ${#company_list[*]}"
271     echo "  Number of matching records: $matches"
272     echo
273     echo "                                ISSUES"
274     echo "                                -----"
275     for i in ${issue_list[*]};
276     do
277         i=$(echo $i | tr "_" " ")
278         echo $i
279     done
```

```
280     echo
281     read -p "Press enter to continue" dog
282
283     done #end of while true
284 }
285
286
287
288 #setup
289
290
291 while true;
292 do
293     clear
294     echo "----- MAIN MENU -----"
295
296     Please select from the following options:
297
298     1.  Process Complaint Files
299     2.  Remove Duplicate Complaint Records
300     3.  Report by Product
301     4.  Exit
302     "
303
304     read -p "Option#: " user_menu_choice
305
306     if [[ user_menu_choice -eq 1 ]];then
307         option_1
308         continue
309     elif [[ user_menu_choice -eq 2 ]];then
310         option_2
311         continue
312     elif [[ user_menu_choice -eq 3 ]];then
313         option_3
314         continue
315     elif [[ user_menu_choice -eq 4 ]];then
316         clear
317         break
318     else
319         clear
320         echo "That is not a valid option.  Please press Enter to try again."
321         read input
322         clear
323         continue
324     fi
325 done
326
```