```sh
#! /bin/sh

#declarations
ZIP_ARCHIVES="Complaints/Zip_Archives"
FILE_ARCHIVES="Complaints/File_Archives"
PROCESSING="Complaints/Processing"
MAIN_DIRECTORY="Complaints"
MASTER_FILE="Complaints/Complaints_Master.csv"

#ensure in home directory
cd ~

setup(){

    #make or remake the main folder structure
    if [[ -d $MAIN_DIRECTORY ]];then
        rm -r $MAIN_DIRECTORY
        mkdir -p "$FILE_ARCHIVES"
        mkdir -p "$ZIP_ARCHIVES"
        mkdir -p "$PROCESSING"
    else
        mkdir -p "$FILE_ARCHIVES"
        mkdir -p "$ZIP_ARCHIVES"
        mkdir -p "$PROCESSING"
    fi

    #setup master file with header
    if ! [[ -f $MASTER_FILE ]];then
        echo "Complaint ID,Date Received,Company,Product,Issue" > $MASTER_FILE

    fi

}

option_1(){

    #tar -xzf aggregate_complaints_001.tar.gz -C $FILE_ARCHIVES #THIS IS FOR TESTING ONLY

    #create a list of all zip files to process
    active_zip_file_list=("aggregate_complaints"*)

    #unpack each zip file then move the zip file to the zip archive
    for i in ${active_zip_file_list[*]};
        do
            tar -xzf $i -C "$PROCESSING"
            cp $i "$ZIP_ARCHIVES/$i"
        done

    #notify the user of a wait
    echo "Please wait, processing "${#active_zip_file_list[*]}" zipped complaint file(s)"

    #populate arrays of file names in processing and in file archives WARNING: THESE ARE FULL PATHS
    active_file_list=("$PROCESSING"/*)
    archive_file_list=("$FILE_ARCHIVES"/*)

    #ensure that we don't process a file that has previously been processed
    for i in ${active_file_list[*]};
        do
            for j in ${archive_file_list[*]};
                do
                    i=($(echo $i | cut -d "/" -f 3))    #extracting just the file name
                    j=($(echo $j | cut -d "/" -f 3))    #extracting just the file name
                    if [[ $i == $j ]];then
                    rm $PROCESSING/$i                   #keep in mind that $i is now just the file name
                    echo $i" deleted"
                    fi
                done
        done


    #process all files in processing, move them to file archives as they are processed
    #issues: skipping first line, last line maybe
    active_file_list=("$PROCESSING"/*)
    for i in ${active_file_list[*]};
    do
        #i=($(echo $i | cut -d "/" -f 3))    #extracting just the file name
        #echo $i

        while IFS= read -r line
        do
            linelen=${#line}
            if [[ linelen -eq 2 ]];then     #this should work to catch the last line as well
                continue
            fi
            line=$(echo $line | tr " " "_")
            mya=($(echo $line | cut -d ":" -f 1- --output-delimiter=" "))

            #get complaint id
            cid=${mya[-1]}
            cid=$(echo $cid | cut -d "\"" -f2)

            #get date received
            date_received=${mya[1]}
            date_received=$(echo $date_received | cut -d "\"" -f2)

            #get company WARNING company is messed up
            company=${mya[ -11]}
            company=$(echo $company | tr "," "_")
            company=$(echo $company | cut -d "\"" -f2)

            #get product
            product=${mya[2]}
            product=$(echo $product | tr "," "_")
            product=$(echo $product | cut -d "\"" -f2)


            #get issue
            issue=${mya[4]}
            issue=$(echo $issue | tr "," "_")
            issue=$(echo $issue | cut -d "\"" -f2)

            #write the data to the file
            output="$cid,$date_received,$company,$product,$issue"
            echo $output >> $MASTER_FILE

            #read < /dev/tty TESTING PURPOSES ONLY
        done < $i

        #move the processed file
        i=($(echo $i | cut -d "/" -f 3))
        mv "$PROCESSING/$i" "$FILE_ARCHIVES/$i"

    done

}

option_2(){
    #cleanup

    #load an array of all file records
    while IFS= read -r line
        do
            #echo $line
            if [[ $line == "Complaint ID,Date Received,Company,Product,Issue" ]];then     #skip the header
                continue
            fi

            cid_list+=($(echo $line | cut -f 1-))

            #read < /dev/tty TESTING PURPOSES ONLY
        done < $MASTER_FILE

        current_count=${#cid_list[*]}
        echo "Number of current records: $current_count"

        cid_list=($(
            for i in ${cid_list[*]};
            do
                echo $i
            done | sort -u))

        post_count=${#cid_list[*]}
        echo "Number of records after removing duplicates: $post_count"
        echo "Duplicate records removed: $(( $current_count - $post_count ))"

        #now write that new list to the csv file (but use a duplicate for testing)
        echo "Complaint ID,Date Received,Company,Product,Issue" > "$MASTER_FILE"
        for i in ${cid_list[*]};
        do
            echo $i >> "$MASTER_FILE"
        done

        read -p "Press Enter to continue: "

}

option_3(){
    #reporting
    clear
    #initialize list and load all records into csv_loaded list
    while IFS= read -r line
        do
            #echo $line
            if [[ $line == "Complaint ID,Date Received,Company,Product,Issue" ]];then     #skip the header
                continue
            fi

            csv_loaded+=($(echo $line | cut -f 1-))

            #read < /dev/tty TESTING PURPOSES ONLY
        done < $MASTER_FILE

    #load products into product_list
    for i in ${csv_loaded[*]};
    do
        product=$(echo $i| cut -d "," -f4)
        product_list+=($product)
    done

    #solicit products and show reporting
    while true;
    do
        matches=1
        issue_list=()
        #product_list=()
        company_list=()
        clear
        echo "AVAILABLE PRODUCTS"
        echo "------------------"

        #remove duplicate products
        product_list=($(
            for i in ${product_list[*]};
            do
                echo $i
            done | sort -u))

        #list the products
        for i in ${!product_list[*]};
        do
            i=$(( $i + 1 ))
            product=$(echo ${product_list[$i-1]} | tr "_" " ")
            product=$(echo $product | sed 's/  / /g')
            if [[ $i -lt 10 ]];then
                echo " $i  $product"
            else
                echo "$i  $product"
            fi
        done

        #get the user product number choice
        echo
        read -p "Enter the product number (zero to exit): " choice
        my_count=${#product_list[*]}
        #error trapping for correct input
        #echo $choice
        if [[ $choice -eq 0 ]];then
            break
        fi

        #get the proper product from the product list, based on the user's
```

```
numerical input
        choice_text=${product_list[$choice -1]}

        #get the matching issues and companies
        for i in ${csv_loaded[*]};
        do
            product=$(echo $i | cut -d "," -f4)
            if [[ $product == $choice_text* ]];then
                issue=$(echo $i | cut -d "," -f5)
                issue_list+=($issue)
                company=$(echo $i | cut -d "," -f3)
                company_list+=($company)
            fi
        done

        #get a count of matching records (issue list length with dupes)
        matches=${#issue_list[*]}

        #remove duplicates from the issue list
        issue_list=($(
        for i in ${issue_list[*]};
        do
            echo $i
        done | sort -u))

        #remove duplicates from the company list
        company_list=($(
        for i in ${company_list[*]};
        do
            echo $i
        done | sort -u))

        #show the report elements
        clear
        product=$(echo $choice_text | tr [:lower:] [:upper:])
        choice_text=$(echo $choice_text | tr [:lower:] [:upper:])
        choice_text=$(echo $choice_text | tr "_" " ")
        choice_text=$(echo $choice_text | sed 's/  / /g')
        echo "PRODUCT: $choice_text"
        echo "Number of companies involved: ${#company_list[*]}"
        echo "  Number of matching records: $matches"
        echo
        echo "                      ISSUES"
        echo "                      ------"
        for i in ${issue_list[*]};
        do
            i=$(echo $i | tr "_" " ")
            echo $i
        done
        echo
        read -p "Press enter to continue" dog

    done #end of while true
}



#setup


while true;
do
    clear
    echo "------ MAIN MENU ------

Please select from the following options:

1.  Process Complaint Files
2.  Remove Duplicate Complaint Records
3.  Report by Product
4.  Exit
"

    read -p "Option#: " user_menu_choice

    if [[ user_menu_choice -eq 1 ]];then
        option_1
        continue
    elif [[ user_menu_choice -eq 2 ]];then
        option_2
        continue
    elif [[ user_menu_choice -eq 3 ]];then
        option_3
        continue
    elif [[ user_menu_choice -eq 4 ]];then
        clear
        break
    else
        clear
        echo "That is not a valid option.  Please press Enter to try again."
        read input
        clear
        continue
    fi
done

```