Feedback on Prototype for Trin

General

1. Have one program that acts as an installation program for all your files.
2. Create a colour scheme reflective of the company ethos.

Staff

1. Have an emergency contact for the member of staff.
2. Make a clear prompt as to how you want the format of the NI no entered.
3. Will adding staff have restricted access (as it involves user privledges?).
4. On new staff I would not display level of access
5. When changing staff display their current details so you can verify
6. When deleting staff, display their name so you can verify it's the right one.

Cushnet.

1. Add the field "title" to customer.
2. Some idea for change
3. Some idea for delete.

Stock.

1. On your SOA stock include descriptions of the stock on the output.
2. When viewing stock by stock ref I would show all the fields.

Tran ②

Stock contd

3. Some feedback for change
4. Some feedback for delete.
5. Introduce a minimum stock level and search for stock to be reordered.

Quotes.

1. When viewing a quote introduce £ signs and set up tabs to align quote better.
2. Include the date the quote was prepared.
3. Some on change
4. Some on delete.

Booking/scheduling.

1. When prompted to enter the day (it says date).
2. When I view the schedule it has the days of the week but no reference to any dates (was this just part of the prototyping) If not there needs to be referencing to dates
3. a) When adding a booking I should add a date and it convert to the day automatically
   b) Do I have to enter each hour seperately?
4. (a) When viewing it gives me the reference but will I need someway of knowing where I am going in the week? (like cross referencing)

Fran 3.

Why do you have "clear all schedule"?

Invoices

1. Should the option to "add invoice" be automatic once a job is complete? Would it refer back to the quote at all?
2. Should a date of invoice be included?
3. How long is given for an invoice to be paid? Should you be searching for unpaid invoices?
4. Same for change?
5. Same for delete?
6. Would you include VAT seperately in the final calculation?

GENERAL:			
Feedback Point	Accept / Reject	Justification	Re-design requirements
Have one program that acts as an instillation program for all your files	Accept	I believe that the files should all be in one program to help make the installation easier and quicker for the user.	None
Create a colour scheme reflective of the company ethos	Accept	I think that a colour scheme is a great way to represent the company throughout the system.	Change design output to show red lettering on the menus.
STAFF:			
Feedback Point	Accept / Reject	Justification	Re-design requirements
Have an emergency contact for the member of staff	Accept	I think that I should accept this because an emergency number is important for the company to have in case a member of staff suddenly becomes ill or something happens.	Change design input grid to allow a user to enter an emergency mobile number. Change pseudocode to allow number to be entered and validated. Change validation sheet to show validation of this Update inputs to show extra field.
Make a clear prompt as to how you would want the format of the NI	Accept	I think I should accept this because if the user enters it in a different format it may go over the character limit and will not allow them to store it further and it will be lost so it is important the user knows exactly what to do.	Change design input grid to show example format next to input

Will adding staff have restricted access (as it involves user privileges)?	Reject	I have rejected this because there is no security within the prototype as shown in the prototype justifications so therefore I have rejected it.	None
On view staff I would not display level of access	Accept	I have accepted this because level of access is not necessary to be viewed since only the boss and the employee need to know what the level of access is and if it can be viewed it could be a security issue.	Remove level of access from the view function Remove level of access from the output grid and pseudocode
When changing staff display their current details so you can verify	Accept	I have accepted this because the user only enters a reference so if they mistype or are unsure on the staff reference the name will appear to ensure it is the correct staff member before they make any changes.	Update pseudocode
When deleting staff display their name so you can verify it's the right one.	Accept	I have accepted this because the user only enters a reference so if they mistype or are unsure on the staff reference the name will appear to ensure it is the correct staff member before they are deleted.	Update pseudocode
CUSTOMER:			
Feedback Point	Accept / Reject	Justification	Re-design requirements
Add the field 'title' to customers	Accept	I have accepted this because it is a norm to include the title when storing details regarding someone so it is important the company also follows these.	Change design input to allow a user to enter a title Change pseudocode to allow title to be entered and validated. Update inputs to show extra field

When changing customer information display their current details so you can verify	Accept	I have accepted this because the user only enters a reference so if they mistype or are unsure on the reference the name will appear to ensure it is the correct customer before they make any changes.	Update pseudocode
When deleting a customer display their current details so you can verify	Accept	I have accepted this because the user only enters a reference so if they mistype or are unsure on the reference the name will appear to ensure it is the correct customer before they are deleted.	Update pseudocode
STOCK:			
Feedback Point	Accept / Reject	Justification	Re-design requirements
On the sort stock include descriptions of the stock on the output	Accept	I have accepted this because the user may not remember what the reference represents so by showing further information it will help them know what they need to re-order	Amend design output to show extra fields such as ‘colour’ and ‘type’.
When viewing stock by stock ref show all fields	Accept	I have accepted this because when a user views stock they be searching to know a specific piece of information so by outputting all information then they know they will find what they need rather than only showing some fields.	Amend design output to include all fields. Amend pseudocode to show the output of all fields.
When changing stock information display the current details so you can verify	Accept	I have accepted this because the user only enters a reference so if they mistype or are unsure on the reference more information regarding that reference such as	Update pseudocode

		colour and type will appear to ensure it is the correct item of stock before they make any changes.	
When deleting an item of stock display the current details so you can verify	Accept	I have accepted this because the user only enters a reference so if they mistype or are unsure on the reference the colour and type of the item of stock will appear to ensure it is the correct item of stock before they are deleted.	Update pseudocode
Introduce a minimum stock level and search for stock to be reordered	Accept	I have accepted this because it is important the staff know what stock is low since it is most likely to be more popular types of paint which are used more this means that they may need it again for an upcoming job so will need to know if they should order it in preparation. Furthermore, it also helps to keep the company organised since if once stock is low more is reordered they are never without.	Add new option to stock menu of 'Stock to be reordered' to design output grid Include new search in pseudocode.
QUOTES:			
Feedback Point	Accept / Reject	Justification	Re-design requirements
When viewing a quote introduce £ signs and set up tabs to align quote better	Accept	I have accepted this because it helps makes the quote clearer to read since there are different numbers regarding different aspect of the quote so by using the £ sign it shows where the prices are shown on the	Amend design outputs and pseudocode to show £ signs.

		quote so it cannot be confused by references etc.	
Include the date the quote was prepared	Accept	I have accepted this because it helps keep the company records organised since when they search for a quote they know if it is old and could be potentially deleted or if it newer and still needed on the system	Amend design inputs to include a date to be entered for when the quote was made. Add input to the pseudocode and show validation Update validation sheet Update input/output sheets
When changing quote information display the current details so you can verify	Accept	I have accepted this because the user only enters a reference so if they mistype or are unsure on the reference more information regarding that quote will appear to ensure it is the correct one before they make any changes.	Update pseudocode
When deleting a quote display the current details so you can verify	Accept	I have accepted this because the user only enters a reference so if they mistype or are unsure on the reference more information regarding that quote will appear to ensure it is the correct one before being deleted.	Update pseudocode
BOOKINGS / SCHEDULES			
Feedback Point	Accept / Reject	Justification	Re-design requirements
When prompted to enter the day it says date	Accept	I have accepted this because it is an error which could be misleading for the user.	Change design output grid to say ‘day’ rather than date.

When I view the schedule it has the days of the week but no reference to any dates	N/A	This does not apply as viewing the schedule by days was only for the prototype.	None
When adding a booking it should add a date and it connect to the days automatically	N/A	Prototype does not include dates.	None
Do you have to enter each hour separately	Accept	I have accepted this because it means that it takes less time for the user to input information since if a job was a week long it would take a lot of a time to add to the schedule.	Amend schedule pseudocode. Amend design input grid.
When viewing it gives the reference but will I need some way of knowing where I am going for the week	Partially Accept	I have accepted this because it will be more useful for the user as it helps them be more efficient rather than then having to also search for the address however I will not be able to show the address on the schedule since it would not fit so instead I will create an extra menu option allowing the user to see the staff members work for just that day in a list format so the address can also be shown.	Amend pseudocode Amend design output grid
Why do you have 'clear all schedule'?	Accept	The schedule works in advance so you do not need to clear it as you would clear future bookings from the schedule.	Remove from pseudocode
INVOICES:			
Feedback Point	Accept / Reject	Justification	Re-design requirements

Should the option to 'add invoice' be automatic once a job is complete? Would it refer back to the quote at all	Partially accept	I think that most of the information can be linked from the quote to save time apart from the dates worked as this may change due to unforeseen circumstances however everything else can be linked.	Update pseudocode to show linking of quote file and removal of fields Update design input grid to remove fields that are no longer needed Update input/output sheets Update validation sheet
Should a date for the invoice be included?	Accept	I have accepted this because it is important for the staff to know when the invoice was sent to help manage payments and if the customer has still not paid, they can compare it to the date the invoice was given.	Add new field 'dateofinvoice' Update pseudocode Update design input grid Update input / output sheet Update validation sheet
How long is given for all invoices to be paid?	Reject	I do not think this is necessary as it is only a small company so the staff will be able to keep the time of invoices to be paid under control since there is no time limit in place within the original system as most customers pay straight away.	None
Should you be searching for unpaid invoices?	Reject	I believe that we should search for unpaid invoices because it helps the staff feel organised and is an efficient way to contact customers if needed.	None
When changing invoice information display the current details so you can verify	Accept	I have accepted this because the user only enters a reference so if they mistype or are unsure on the reference more information regarding that invoice will appear to ensure it is the correct one before they make any changes.	Update pseudocode

When deleting an invoice display the current details so you can verify	Accept	I have accepted this because the user only enters a reference so if they mistype or are unsure on the reference more information regarding that invoice will appear to ensure it is the correct one before being deleted.	Update pseudocode
Would you include VAT separately in the final calculation?	Reject	I have rejected this because VAT is a standard shown across all businesses, so it is important this company also sticks to those standards by showing the VAT.	None

Variable Tables

Staff:

Field Name	Description	Type	Length	Sample	Validation Type	Further description	Expected Output
staffref	Unique ID for each member of staff. Key Field.	Int	Short	2	File Check	Read back the file to check if the staffref entered is new or whether it already exists within the file as all staff references must be unique on the system, cannot be duplicates.	This staff reference already exists. Please re-enter a different number.
fnamestaff	Staff members first name.	Char	15	Stuart	Presence	Makes sure that data has been entered to record to the file.	Ensure that you have entered a first name.
lnamestaff	Staff members last name.	Char	15	Watson	Presence	Makes sure that data has been entered to record to the file.	Ensure that you have entered a last name.
oneadstaff	1st line of a staff members home address.	Char	30	15 Cherry Tree Close	Presence	Makes sure that data has been entered to record to the file.	Please check to see if line 1 of the address has been entered.
twoadstaff	2nd line of a staff members home address.	Char	15	Kiveton Park	Presence	Makes sure that data has been entered to record to the file.	Please check to see if line 2 of the address has been entered. If it is not applicable please

							use ‘N/A’.
threeadstaff	3rd line of a staff members home address.	Char	15	Sheffield	Presence	Makes sure that data has been entered to record to the file.	Please check to see if line 3 of the address has been entered.
pcodestaff	The postcode for the staff members home address	Char	8	S26 5QQ	Format	AA9A 9AA A9A 9AA A9 9AA A99 9AA AA9 9AA AA99 9AA Validates the postcode for all UK formats checking the length and that all letters and digits are in the correct space for the corresponding length.	Please check that the letters are in the correct place. Please check that the numbers are in the correct place. Please make sure that the postcode is of the correct length including spaces.
telnostaff	The contact number for the member of staff.	Char	11	07743117706	Format	07NNNNNNNN Validates a UK mobile number ensuring it starts with a zero and is exactly 11 digits long and are all digits.	Please make sure that the mobile number entered begins with 07. Please make sure that the mobile number entered is exactly

							11 digits long.
emtel	The emergency mobile number for the member of staff	Char	11	07743117706	Format	07NNNNNNNN Validates a UK mobile number ensuring it starts with a zero and is exactly 11 digits long and are all digits.	Please make sure that the mobile number entered begins with 07. Please make sure that the mobile number entered is exactly 11 digits long.
ninum	The national insurance number for the member of staff.	Char	9	QQ 12 34 56 A	Format	LL NNNNNN L Validates a national insurance number ensuring it is of the correct length, with all spaces, digits and letters in the correct place.	Please make sure that the NI number is 13 characters in length including spaces. Please make sure that there are the correct spaces within the NI number. Please make sure that the letters

							are typed correctly an in the right place. Please make sure that the numbers are typed correctly an in the right place.
username	The staff members unique login for the system	char	20	TSmith	Unique File Check	Read back the file to check if the username entered is new or whether it already exists within the file.	This username is already in use. Pleas try again with a different username.
password	Staff members unique password for logging onto the system.	Char	15	TylerSm!th41	Password criteria check	Validates that the new password entered is of at least 8 characters and has either a number or piece of punctuation within it.	Please make sure that the password is at least 8 characters long. Please make sure that the password has either a piece of

							punctuation or a number.
Loa	Hierarchical access for the staff member to allow them to do certain administration roles within the company.	int	1	3	Range	Makes sure that the level of access is between 1-3 as these are the levels I have chosen. 3 being the most access. 1 being the least.	Please make sure the level of access is between 1-3.

Customers:

Field Name	Description	Type	Length	Sample	Validation Type	Further Description	Expected Output
custref	Unique ID for each customer. Key Field.	Int	Short	5	File Check	Read back the file to check if the custref entered is new or whether it already exists within the file as all customer references must be unique on the system, cannot be duplicates.	This customer reference already exists. Please re-enter a different number.
title	Title for the customer	Char	10	Miss	Presence	Makes sure that data has been entered to record to the file.	Ensure that you have entered a title.
fnamecust	Customers first name.	Char	15	Emily	Presence	Makes sure that data has been entered to record to the file.	Ensure that you have entered a

							first name.
lnamecust	Customers last name.	Char	15	Carlton	Presence	Makes sure that data has been entered to record to the file.	Ensure that you have entered a last name.
1adcust	1st line of a customer's home address.	Char	30	26 Wiltshire Bank	Presence	Makes sure that data has been entered to record to the file.	Please check to see if line 1 of the address has been entered.
2adcust	2nd line of a customer's home address.	Char	15	Sothall	Presence	Makes sure that data has been entered to record to the file.	Please check to see if line 2 of the address has been entered. If it is not applicable please use 'N/A'.
3adcust	3rd line of a staff members home address.	Char	15	Sheffield	Presence	Makes sure that data has been entered to record to the file.	Please check to see if line 3 of the address has been entered.
pcodecust	The postcode for the customers home address	Char	8	S20 2NT	Format	AA9A 9AA A9A 9AA A9 9AA A99 9AA AA9 9AA AA99 9AA Validates the postcode for all UK formats checking the length and that all letters and digits are in the correct space for the	Please check that the letters are in the correct place. Please check that the numbers are in the correct place. Please make sure that

						corresponding length.	the postcode is of the correct length including spaces.
telnocust	The contact number for the customer.	Char	11	07795236573	Format	0NNNNNNNNN Validates a UK mobile number ensuring it starts with a zero and is exactly 11 digits long and are all digits.	Please make sure that the mobile number entered begins with a 0. Please make sure that the mobile number entered is exactly 11 digits long.

Quotes:

Field Name	Description	Type	Length	Sample	Validation Type	Further Description	Expected Output
quoteref	Unique ID for each quote. Key Field.	Int	Short	3	File Check	Read back the file to check if the quoteref entered is new or whether it already exists within the file as all quote references must be unique on the system,	This quote reference already exists. Please re-enter a different number.

						cannot be duplicates.	
custno	Unique ID for the customer relating to the job. Foreign Key.	Int	Short	5	File Check	Read back the file to check if the custno entered is new or whether it already exists within the file.	This customer number already exists. Please re-enter a different number.
quotedate	Stores the date that the quote was produced for the customer	Char	10	20/06/2023	Format DD/MM/YYYY Range check – using systems clock	Ensures that the date entered follows a particular format as represented. Checking that the dates are correct for the corresponding month. That the forward slashes are in the correct place. Checks the months are in the range 01-12. Also checks that the date entered is in the future and won't allow a date from the past to be used by comparing to the systems clock.	Please ensure that the dates are between the 1st and 31st for this month. Please ensure the dates are between the 1st and 30th for this month. Please ensure the dates are between the 1st and 28th for this month. Please ensure the dates are between the 1st and 29th for this month.

							Please check the month is in the range 01-12. Please check that the forward slashes are in the correct place. Please check that date entered is after the current date.
mainjobdesc	An abbreviated description of the job requirements.	Char	50	Bedroom Strip WP Paint ceil and walls	Presence	Makes sure that data has been entered to record to the file.	Please ensure you have entered a description of the job for the quote.
eststock	An estimated idea of what stock will be required for the job.	Int	Short	1 3 5	Presence	Makes sure that data has been entered to record to the file.	Please ensure you have entered the stock ID's for the stock you think you may require.
qnumofdays	To produce the quote calculation the estimated number of days required to	Int	Short	7	Range	Makes sure that the data entered is appropriate in context to the variable – ensure user does	Please ensure you have entered the correct number of days

	complete the job is used.					not enter 70 days instead of 7 etc.	you expect to work.
matprices	The price of the materials required for the job. This will be linked automatically from the stock file so will have already been validated upon entry of the item of stock.	Int	Short	100			
Mileage	Cost of fuel for travelling to a job each day.	Int	Short	60	Presence	Makes sure that data has been entered to allow a calculation for the job to happen.	Please ensure you have entered the milage costs.
totalprice	The final price of the job all added together. This will be outputted to the screen not entered so cannot be validated upon entry.	Int	Short	710			

Stock:

Field Name	Description	Type	Length	Sample	Validation Type	Further Description	Expected Output
stockref	Unique ID for each	Int	Short	3	File Check	Read back the file to	This stock reference

	item of stock. Key field.					check if the stockref entered is new or whether it already exists within the file as all stock references must be unique on the system, cannot be duplicates.	already exists. Please re-enter a different number.
quantity	The amount of each item of stock currently stored at that time.	Int	Short	5	Range check	Makes sure that the data entered is appropriate in context to the variable – eg quantity entered is 10 and not 100	Please ensure you have entered the quantity of this item you currently have correctly.
colour	The colour of the paint.	Char	15	Satin White	Presence	Makes sure that data has been entered to record to the file.	Please ensure that the colour of the paint has been entered. If colour isn't applicable please use N/A
volume	The volume of the tin of paint in litres.	Int	Short	50	Range	Makes sure that the data entered is appropriate in context to the variable – eg volume entered is	Please ensure that the volume of the paint entered is correct.

						10l not 100l	
type	The category of paint used.	Char	10	Matte	Presence	Makes sure that data has been entered to record to the file.	Please ensure that the type of the paint has been entered. If type of paint isn't applicable please use N/A
stockprice	The cost of the item of stock	Int	short	96	Range	Ensure the price of the stock entered is a suitable value and has not been mistyped.	Please ensure the price entered is correct.

Invoices:

Field Name	Description	Type	Length	Sample	Validation Type	Further Description	Expected Output
invoiceref	Unique ID for each invoice. Key field.	Int	Short	3	File Check	Read back the file to check if the invoiceref entered is new or whether it already exists within the file as all invoice references must be unique on the system, cannot be duplicates.	This invoice reference already exists. Please re-enter a different number.
custno	The unqiue identifier	Int	Short	3	File Check	Read back the file to	This customer

	of customer. Foreign key					check if the custno entered is new or whether it already exists within the file as all customer references must be unique on the system, cannot be duplicates.	number already exists. Please re-enter a different number.
invoicedate	Stores the date that the invoice was produced for the customer	Char	10	20/06/2023	Format DD/MM/YYYY Range check – using systems clock	Ensures that the date entered follows a particular format as represented. Checking that the dates are correct for the corresponding month. That the forward slashes are in the correct place. Checks the months are in the range 01-12. Also checks that the date entered is in the future and won't allow a date from the past to be used by comparing to the systems clock.	Please ensure that the dates are between the 1st and 31st for this month. Please ensure the dates are between the 1st and 30th for this month. Please ensure the dates are between the 1st and 28th for this month. Please ensure the dates are between the 1st

							and 29th for this month. Please check the month is in the range 01-12. Please check that the forward slashes are in the correct place. Please check that date entered is after the current date.
jobstartdate	The date that work on the job began.	Char	10	20/06/2023	Format DD/MM/YYYY Range check – using systems clock	Ensures that the date entered follows a particular format as represented. Checking that the dates are correct for the corresponding month. That the forward slashes are in the correct place. Checks the months are in the range 01-12.	Please ensure that the dates are between the 1st and 31st for this month. Please ensure the dates are between the 1st and 30th for this month. Please ensure the dates are

						Also checks that the date entered is in the future and won't allow a date from the past to be used by comparing to the systems clock.	between the 1st and 28th for this month. Please ensure the dates are between the 1st and 29th for this month. Please check the month is in the range 01-12. Please check that the forward slashes are in the correct place. Please check that date entered is after the current date.
jobenddate	The final date of the job.	Char	10	27/06/2023	Format DD/MM/YYYY Range check	Ensures that the date entered follows a particular format as represented. Checking that the dates are correct for the	Please ensure that the dates are between the 1st and 31st for this month. Please ensure

						correspondin g month. That the forward slashes are in the correct place. Checks the months are in the range 01-12. Checks that the final date entered is later than the start date entered	the dates are between the 1st and 30th for this month. Please ensure the dates are between the 1st and 28th for this month. Please ensure the dates are between the 1st and 29th for this month. Please check the month is in the range 01-12. Please check that the forward slashes are in the correct place. Please check that the final day is after the start date.

finaldesc	Abbreviated description of the job after it has been completed.	Char	50	Bedroom Strip WP Paint ceil and walls	Presence	Makes sure that data has been entered to record to the file.	Please ensure you have entered a description of the job for the invoice.
inumofdays	The number of days that the job took from the start date to the final date.	Int	Short	7	Presence	Makes sure that data has been entered to allow a calculation for the job to happen.	Please ensure you have entered the number of days you worked.
matprices	The price of the materials paid for during the duration of the job.	Int	Short	100	Range	Makes sure that data has been entered to allow a calculation for the job to happen.	Please ensure you have entered the price of materials.
Mileage	Cost of fuel for travelling to a job each day.	Int	Short	60	Presence	Makes sure that data has been entered to allow a calculation for the job to happen.	Please ensure you have entered the milage costs.
totalprice	The final price of the job all added together. This will be calculated within the program and is not entered by the user and therefore cannot be validated	Float	4.2	0590.35			

	upon entry.						
paid	Identifies whether or not the invoice has been paid or not.	Int	short	1	Range	Makes sure that data is between 0 and 1. 0 = No 1 = Yes	Please ensure you have entered whether or not the job has been paid for by choosing either Y or N.

Staff member inputs information about a customer, this will then be stored in the customer file

```
Add Customer
**********

Enter new customer reference: 4

Enter first name: Liam
Enter last name : Smith

Enter address line 1: 19 Northlands
Enter address line 2: Hartnill
Enter address line 3: Sheffield
Enter postcode      : S26 7XZ

Enter mobile number: 07700078143
```

A unique reference is key to identifying each customer

Address is required as this is where the job will take place

A contract number will be required if the employee wishes to ask for extra details or confirm a booking etc.

* Enter title: Mr

A search which outputs important information on a specific staff member

View Staff Member by Staff Reference
****

Enter staff reference: 2

Name: Stuart Watson

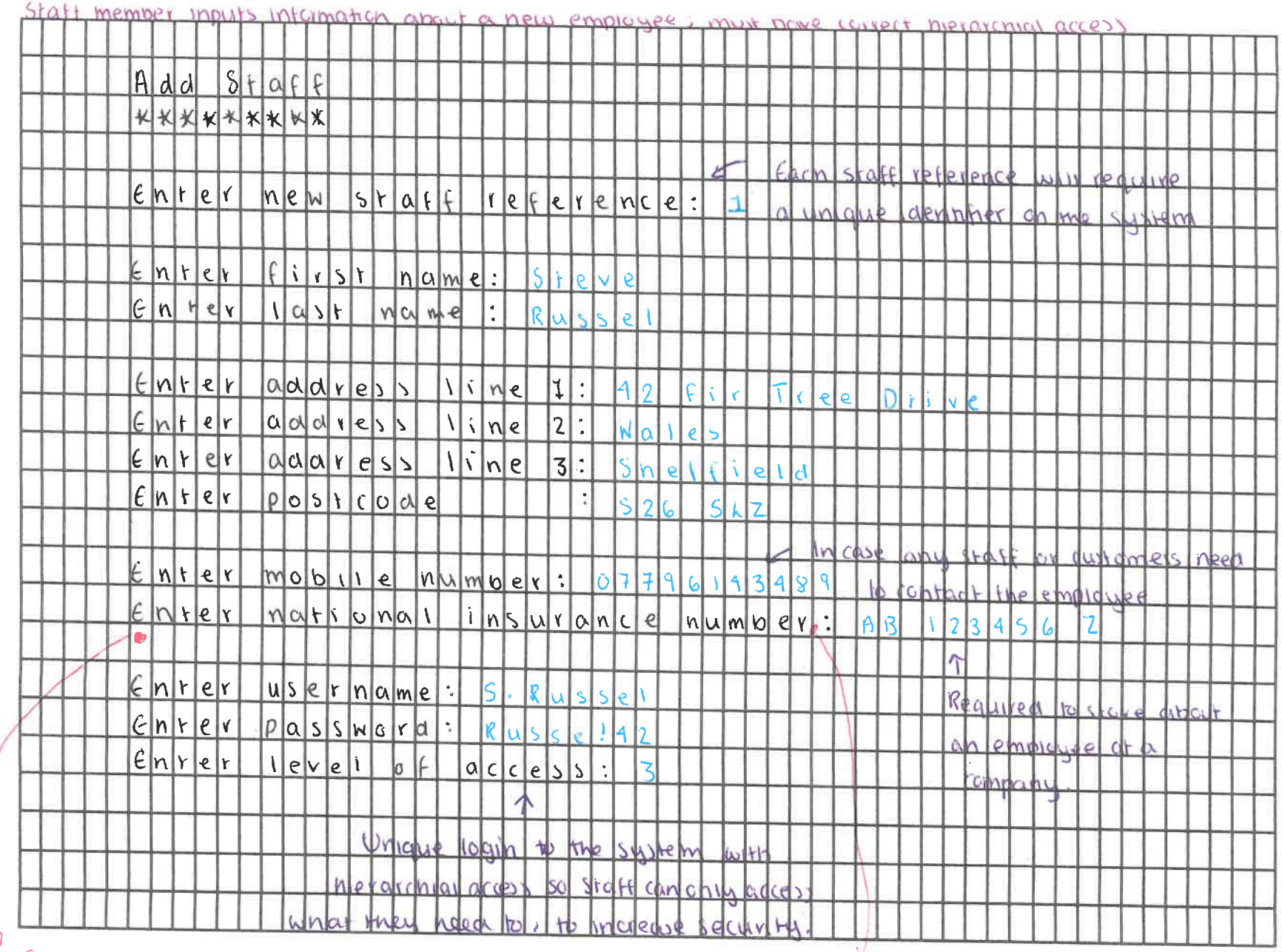
Address: 12 Oak Drive
Kiveton Park
Sheffield
S26 5SQ

Telephone Number: 07719890314 ← Important to output in case the member of staff needs to be contacted immediately

NI Number: QQ 12 34 56 C ← A requirement to have stored about an employee

Emergency telephone number: 07484206960
↑
Important in case something happens to a staff member

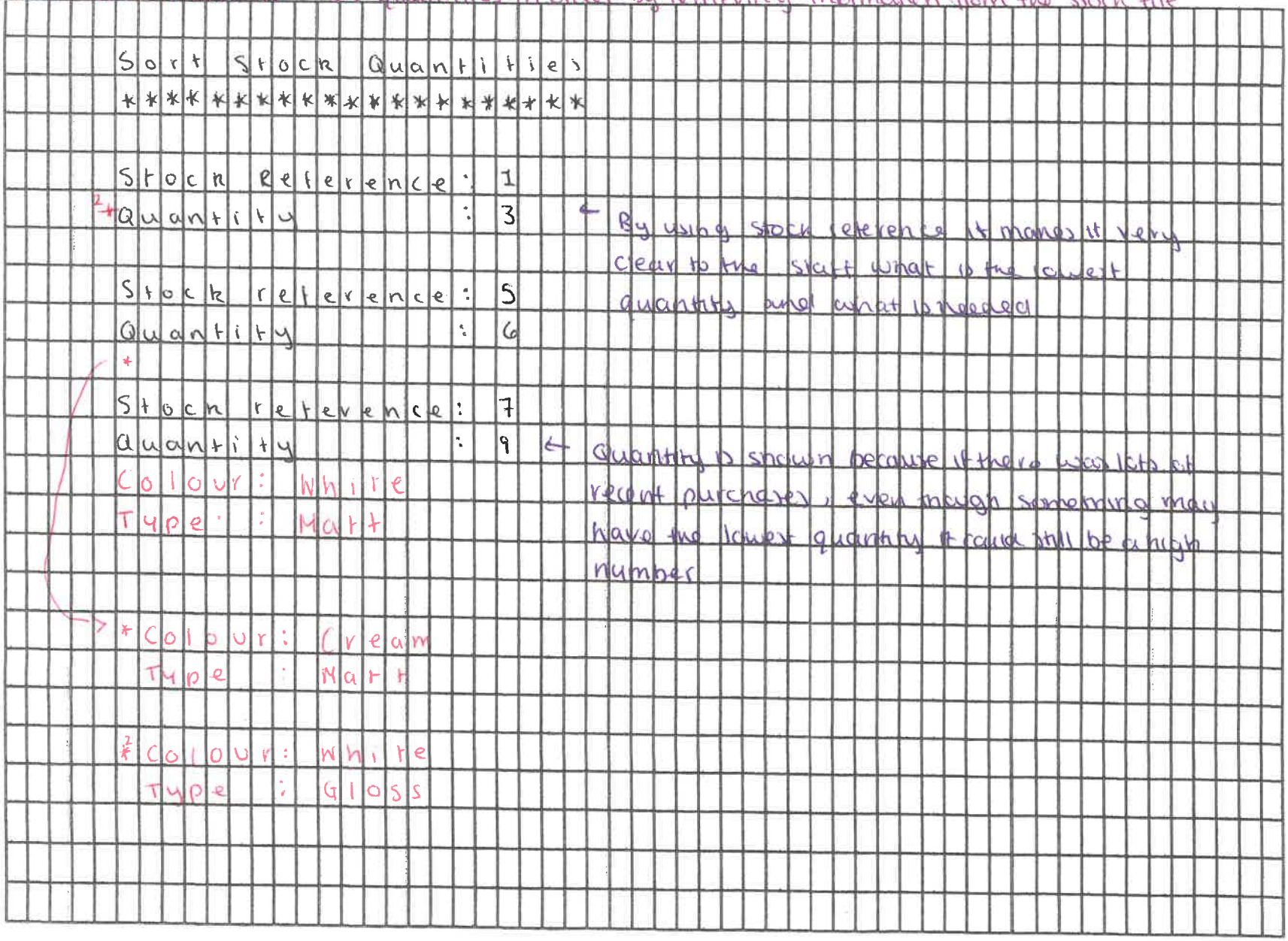
Staff member inputs information about a new employee; must have correct hierarchical access



Enter emergency telephone number:

Enter national insurance number (AB 12 34 56 C):

A sort which shows the stock quantities in order by retrieving information from the stock file



A search which outputs specific information about a piece of stock, to help staff prepare for a job

View Stock by Stock Reference
****

Enter stock reference: 1

Colour: White ← Include colour as the customer will have requested it so the staff will only want to check for the colour they need

Price: 15

Quantity: 4

Checks quantity since if its 0 or 1 staff will need to go and buy some more to be prepared for jobs

So the staff are prepared when they go to buy, but also to help provide a quote / invoice

Type: Malt

Volume: 10L

Staff member enters information on a quote; Stock price is linked automatically from the file to be used in the calculation.

Add Quote
****

Unique ID to identify each quote

Enter quote reference: 4
Enter customer reference: 4 ← Helps link to customer

Job description: PUT
KITCHEN
CRM

← Helps the staff remember the details of the job as it gets closer to the time

*

Enter number of days on the job: 5
Enter travel costs: 60

How many items of stock are required: 4

Enter Stock ID: 2
Enter Stock ID: 4
Enter Stock ID: 3
Enter Stock ID: 5

← So the system can find the price of each item of stock to be used in the calculation

Price Breakdown
****

Materials : £150
Labour : £450
Mileage : £60 ← So the customer has an expectation on what the price will be
VAT : £132
Total : £792

* Enter date quote is produced: 10/12/2023

A search which uses the customer reference to read back against the file

View Quote by Customer Reference
****

Enter customer reference: 4

Customer name: Liam Smith
Telephone number: 07749173406
Quote reference: 4

Job description: PNT
KITCHEN
CRM

← To remind the staff of what they will be doing when they arrive at the job

Date quote was produced: 10/12/2023
Number of days worked: 7 ← So the staff can plan what they will do each day on the job to make sure it is completed on time

Price Breakdown
****

Materials: £90
Labour: £560 ← So the staff are able to answer any enquiries about the price of the job from the customer.
Mileage: £25
VAT: £135
Total: £810

A search which outputs which jobs currently haven't been paid for.

```
View unpaid invoices
********************

Customer reference: 2
Customer reference: 4
Customer reference: 9
Customer name       Niam Smith
Customer telephone: 07853577355

Customer name: Christine Jones
Customer telephone: 07922426829

Customer name      : Stewart Browning
Customer telephone: 07516038616
```

← Outputs the customer reference so that the staff can go and view a customer's information, such as a telephone number

Staff member enters dates, quote ref entered will then link other required information.

```
Add Invoice
**********

Enter invoice reference: 4

Enter customer reference: 1

Enter quote reference: 4

Enter the date the invoice is created: 12/12/2023

Enter start date: 05/12/2023
Enter end date  : 10/12/2023

Is job paid for (0=Yes, 1=No): 1
```

Unique to identify each invoice on the system

← Allows information to be linked rather than re-typing.

Allows unpaid invoices to be found on the system so that the customer can be found in payment is required

Staff member enters booking information below to be added to the schedule

```
Add Booking
**********

Enter staff reference: 1

Enter dayc of the booking(1-7): 2

Enter start time (7-19): 9
Enter finish time(7-19): 17

Enter quote reference: 4
```

So the booking is added to the correct staff members schedule

Inputs the 2 times to ensure all hours in that range are added to the schedule

Inputted to be displayed on the schedule so an employee can then search the quote to find the required information

A search which shows a staff member the job they have to do today with important information

Today's Job:

Date: 12/12/2023
Staff Member: 1

Quote reference: 4
Job Description: Paint, Kitchen, White

Outputted so the user knows what is required for the job when they get there

Customer Reference: 1
Customer name: Miss Carla Ray

Home Address: 17 Oak Close
Sothall
Sheffield
S20 4PQ

Outputted so the user knows where they will be going that day and who it is for.

Random Access – Customers

View customer by customer reference:

```
Proc ViewbyCustNum
ouput “Enter the customer number: “
input custref
Open CustomerFile
Locate(custref)
Output “Customer name: ” , a_cust.titlecust, a_cust.fnamecust a_cust.lnamecust
Output “Address line 1: “ , a_cust.oneadcust
Output “Address line 2: “ , a_cust.twoadcust
Output “Address line 3: “ , a_cust.threeadcust
Output “Postcode: “ , a_cust.pcodecust
Output “Telephone number: “ , a_cust.telnocust
End proc
```

View customer by customer name:

```
Proc ViewbyCustName
output “Enter customer last name: “
input lnamecust
Open CustomerFile
Locate(lnamecust)
For find = 1 to nci
        if (a_cust.lnamecust = lnamecust)
                output “Title: “, a_cust.titlecust
                output “First name: ” , a_cust.fnamecust
                output “Last name: “ , a_cust.lnamecust
                output “Address line 1: “ , a_cust.oneadcust
                output “Address line 2: “ , a_cust.twoadcust
```

```
            output “Address line 3: “ , a_cust.threeadcust
            output “Postcode: “ , a_cust.pcodecust
            output “Telephone number: “ , a_cust.telnocust
        End if
End for
End proc
```

<u>Add Customer:</u>

```
Proc AddCust
prest =0,presf = 0, presl =0, presaone = 0, presatwo = 0, presathree = 0, pcode =0, telnum=0
output “Enter a new customer number: “
input custref
Open CustomerFile
If (flag=0)
        While prest = 0
            Output “Enter title: “
            Input a_cust.titlecust
            Prest = Do Proc PresenceCk(titlecust)
        Endwhile
        While presf = 0
            output “Enter first name: “
            input a_cust.fnamecust
            presf = Do Proc PresenceCk(fnamecust)
        Endwhile
        While presl = 0
            output “Enter last name: “
            input a_cust.lnamecust
            presl = Do Proc PresenceCk(lnamecust)
        Endwhile
        While presaone = 0
            output “Enter address line 1: “
```

```
            input a_cust.oneadcust
            presa1 = Do Proc PresenceCk(oneadcust)
        Endwhile
        While presatwo = 0
            output "Enter address line 2: "
            input a_cust.twoadcust
            presatwo = Do Proc PresenceCk(twoadcust)
        Endwhile
        While presathree = 0
            output "Enter address line 3: "
            input a_cust.threeadcust
            presa3 = Do Proc PresenceCk(threeadcust)
        Endwhile
        While pcode = 0
            output "Enter postcode: "
            input a_cust.pcodecust
            pcode = Do Proc PostcodeVal(pcodecust)
        Endwhile
        While telnum = 0
            output "Enter telephone number: "
            input a_cust.telnocust
            telnum = Do Proc TelFormatCk(telnocust)
        Endwhile
        flag = 1
End if
Else
        Output "Customer reference is already in use"
End proc
```

Delete Customer:

```
Proc DelCustomer
```

```
output “Enter customer reference: “

input custref

Open CustomerFile

Locate (custref)

Output"Full name: “, a_cust.fnamecust, a_cust.lnamecust
Output”Is this the correct staff member”
Input result
        If(result = ‘0’)
                If(flag=1)
                                a_cust.fnamecust = a_cust.fnamecust + 1

                                a_cust.lnamecust = a_cust.lnamecust + 1

                                a_cust.1adcust = a_cust.oneadcust + 1

                                a_cust.2adcust = a_cust.twoadcust + 1

                                a_cust.3adcust = a_cust.threeadcust + 1

                                a_cust.pcodecust = a_cust.pcodecust + 1

                                a_cust.telnocust = a_cust.telnocust + 1

                        End if

                End if

Else

        Output “Customer reference does not exist within the system.”

flag = flag –1

End proc
```

Change customer address:

```
Proc ChangeCustomerH

presa=0, presone=0, presatwo=0, presathree=0, pcode=0

output “Enter customer reference: “

input custref

Open CustomerFile

Locate (custref)

Output"Full name: “, a_cust.fnamecust, a_cust.lnamecust
```

```
Output”Is this the correct staff member”
Input result
        If(result = ‘0’)

                If(flag=1)

                While presaone = 0

                        output “Enter address line 1: “

                        input a_cust.oneadcust

                        presaone = Do Proc PresenceCk(oneadcust)

                Endwhile

                While presatwo = 0

                        output “Enter address line 2: “

                        input a_cust.twoadcust

                        presatwo = Do Proc PresenceCk(twoadcust)

                Endwhile

                While presathree = 0

                        output “Enter address line 3: “

                        input a_cust.threeadcust

                        presathree = Do Proc PresenceCk(threeadcust)

                Endwhile

                While pcode = 0

                        output “Enter postcode: “

                        input a_cust.pcodecust

                        pcode = Do Proc PostcodeVal(pcodecust)

                Endwhile

        End if

End if

Else

        Ouput “Customer reference does not exist on the system.”

End proc
```

Change customer telephone number:

```
Proc ChangeCustTelno

telnum=0

output “Enter customer reference: “

input custref

Open CustomerFile

Locate (custref)

Output"Full name: “, a_cust.fnamecust, a_cust.lnamecust
Output”Is this the correct staff member”
Input result
If(result = ‘0’)
        If(flag=1)
                While telnum = 0

                        output “Enter telephone number: “

                        input a_cust.telnocust

                        telnum = Do Proc TelFormatCk(telnocust)

                Endwhile

        Endif

End if

        Else

        Ouput “Customer reference does not exist on the system.”

End proc
```

Sequential File – Schedule

ReadBackBookingSchedule:

```
Proc ReadBackBookingSchedule
Open BookingScheduleFile
for staffno = 1 to 2
        for date = 1 to 365
                for hour = 1 to 12 (7  to 19)
                        read back bookingsch[staffno][date][hour]
                endfor
        endfor
endfor
Close BookingScheduleFile
End proc
```

ReWriteBookingSchedule:

```
Proc ReWriteBookingSchedule
Open BookingScheduleFile
for staffno = 1 to 2
        for date = 1 to 365
                for hour = 1 to 12
                        rewrite bookingsch[staffno][date][hour]
                endfor
        endfor
endfor
Close BookingScheduleFile
End proc
```

Add Booking:

```
Proc AddBooking
Open BookingScheduleFile
Do Proc ReadBackBookingSchedule
Output “Enter staff reference: “
Input staffno
Output “Enter day of the booking: ”
Input date
Output “Enter the start time for that day: “
Input starthour
Output “Enter the finsih time for that day: “
Input endhour
Output “Enter quote reference: “
Input quoteno
While hour>= starthour and hour<= endhour
        ReWrite Bookingsch([staffno][date][hour], quoteno)
        Do Proc ReWriteBooking Schedule
End while
Close BookingScheduleFile
End proc
```

Delete booking:

```
Proc DeleteBooking
Do proc ReadBackBookingSchedule
Output “Enter staff reference for the booking”
Input staffno
Output “Enter the number of days on the booking: “
Input numday
For del = 1 to numday
```

```
        Output “Enter the day of the booking: “
        Input date
        Output “Enter the start time for that day: “
        Input starthour
        Output “Enter finish time for that day: “
        Input endhour
        While hour>= starthour and hour<= emdhour
                ReWriteBookingSch([staffno][date][hour], “ “)
        End while
End for
End proc
```

View schedule:

```
Proc ViewSchedule
Do proc ReadBackBookingSchedule
Output “Enter the staff reference to view their schedule: “
Input staffno
For date = 1 to 365
        Output days[date]
        For hour = 1 to 12
                output bookingsch[staffno][date][hour]
        End for
End for
End proc
```

Change Schedule dates:

```
Proc ChangeDates
```

```
Do proc ReadBackBookingSchedule
Output “Enter the staff reference of the booking you wish to change”
Input staffno
Output “Enter the number of days on the booking: “
Input numday
For del = 1 to numday
        Output “Enter the date of the booking: “
        Input date
        Output “Enter the start time for that day: “
        Input starthour
        Output “Enter the finish time for that day: “
        Input endhour
        Output “Enter quote reference: “
        Input quoteno
        While hour>= starthour and hour<= endhour
                ReWriteBookingSch([staffno][date][hour], quoteno)
        Endwhile
End for
End proc
```

Serial File – Invoices

ReadBackInvoicesFile:

```
Proc ReadBackInvoicesFile
Open InvoicesFile
Read nii
For count = 1 to nii
        read invoiceref(count)
        read custno(count)
        read quoteno(count)
        read invoicedate(count)
        read jobstartdate(count)
        read jobenddate(count)
        read paid(count)
End for
Close InvoicesFile
End proc
```

ReWriteInvoicesFile:

```
Proc ReWriteInvoicesFile
Open InvoicesFile
Write nii
For count = 1 to nii
        write invoiceref(count)
        write custno(count)
        write quoteno(count)
```

```
        write invoicedate(count)
        write jobstartdate(count)
        write jobenddate(count)
        write paid(count)
End for
Close InvoicesFile
End proc
```

Add Invoice:

```
Proc AddInvoice
uniqueref = 0, uniquecust=0, quote =0, idate=0, startdate=0, datesyst = 0, enddate=0,  rangep = 0
Do Proc ReadBackInvoicesFile
While uniqueref = 0
        output “Enter new invoice reference: ”
        input invoiceref[nii]
        uniqueref = Do Proc UniqueVal(invoiceref[nii])
Endwhile
While uniquecust= 0
        output “Enter customer reference: “
        input custno [nii]
        uniquecust= Do Proc UniqueVal(custno[nii])
Endwhile
While quote = 0
Output “Enter quote reference linked to the booking: “
Input quoteno[nii]
quote = Do Proc UniqueVal(quoteno[nii])
```

```
End while
While idate = 0
Output “Enter the date the invoice was produced: “
Input invoicedate[nii]
        Idate = Do proc DateVal(idate[nii])
End while
While startdate= 0 and datesyst = 0
        output “Enter start date: “
        input jobstartdate[nii]
        startdate= Do Proc DateVal(jobstartdate[nii])
        Datesyst = Do Proc SystemsClock(jobstartdate[nii])
Endwhile
While enddate= 0
        output “Enter end date: “
        input jobenddate[nii]
        enddate= Do Proc DateVal(jobenddate[nii])
Endwhile
While rangep= 0
        output “Is job paid for: ( 0 = Yes,  1 = No) “
        input paid[nii]
        rangep= Do Proc RangeCk(paid[nii])
Endwhile
nii = nii + 1
Do Proc ReWriteInvoiceFile
End proc
```

<u>Delete Invoice:</u>

```
Proc DeleteInvoice
```

```
Do Proc ReadBackInvoicesFile
Output “Enter the invoice reference: “
input invoiceref[nii]
For find = 1 to nii
        if (find[nii] = invoiceref)
                Output"Customer Reference: “, custno[find]
                Do proc LocateCustomer(custno)
                Output “Dates worked: “,jobstartdate[find] “-” ,jobenddate[find]
                Output”Is this the correct invoice”
                Input result
                If(result = ‘0’)
                        for del = find to nii
                                invoiceref[del] = invoiceref[del+1]
                                custno[del] = custno[del+1]
                                quoteno[del] = quoteno[del+1]
                                invoicedate[del] = invoicedate[del+1]
                                jobstartdate[del]=jobstartdat[del+1]
                                jobenddate[del] = jobenddate[del+1]
                                paid[del] = paid[del+1]
                        End for
                End if
        Endif
        Else
        Output “Invoice reference does not exist.”
Endfor
nii = nii –1
Do proc ReWriteInvoiceFile
End proc
```

Change Invoice Payment Status:

```
Proc ChangePaid
rangep= 0
Do proc ReadBackInvoiceFile
output “Enter the invoice reference: “
input invoiceref[nii]
For find = 1 to nii
        If( find[nii] = invoiceref)
                Output"Customer Reference: “, custno[find]
                Do proc LocateCustomer(custno)
                Output “Dates worked: “,jobstartdate[find] “-” ,jobenddate[find]
                Output”Is this the correct invoice: ”
                Input result
                If (result = ‘0’)
                        While rangep= 0
                                output “Enter payment status: “
                                input paid[nii]
                                rangep= Do Proc RangeCk(paid[nii])
                        Endwhile
                End if
        End if
        Else
        Output “Invoice reference does not exist.”

End for
Do proc ReWriteInvoicesFile
End proc
```

View invoice by invoice reference:

```
Prov ViewIRef
output “Enter invoice reference: “
input invoiceref[nii]
Do proc ReadBackInvoicesFile
For find = 1 to nii
        If (find[nii] = invoiceref)
                output “ Customer reference: “, custno[nii]
                Do Proc LocateCustomer(custno)
                output “Date invoice was produced: “, invoicedate[nii]
                Output “Dates worked: “ ,jobstartdate[nii] “-”, jobenddate[nii]
                Output “Quote reference: “, quoteno[nii]
                Do proc LocateQuote(quoteno)
        End if
Else
        Output “Invoice reference does not exist.”
End for
End proc
```

Locate Customer:

```
Proc LocateCustomer(custno)
Locate(custref)
If (flag!=0)
output “Customer name”, a_cust.fnamecust , a_cust.lnamecust
output “Telephone number: ”, a_cust.telnocust
End if
Return custno
```

```
End proc
```

Locate Quote:

```
Proc LocateQuote(quoteno)
ReadBackQuoteFile
For find = 1 to nqi
If (quoteno = quoteref)
        output “Job description “, mainjobdesc[nqi]
        output “ Price Breakdown”
        Output “Materials: “, stockcost[nqi]
        Output “Labour: “, labourq[nqi]
        Output “Mileage: “,mileage[nqi]
        Ouptut “VAT: “, vat[nqi]
        Output “Total cost: “, totalcost[nqi]
    End if
End for
End proc
```

Search for unpaid invoices:

```
Proc SearchUnpaid
ReadBackInvoicesFile
For find = 1 to nii
    (if paid == 0)
        output “Customer Reference”, custno[nii]
        Do Proc LocateCustomer(custno)
    End if
    Else
```

output “All invoices are paid for.”

End for

End proc

Serial File – Quotes

ReadBackStaffFile:

```
Proc ReadBackQuotesFile
Open QuotesFile
Read nqi
For count = 1 to nqi
        read quoteref(count)
        read custno(count)
        read quotedate(count)
        read mainjobdesc(count)
        read eststock(count)
        read qnumofdays(count)
        read matprices(count)
        read totalprice(count)
End for
Close QuotesFile
End proc
```

ReWriteQuotesFile:

```
Proc ReWriteQuotesFile
Open QuotesFile
Write nqi
For count = 1 to nqi
        write quoteref(count)
        write custno(count)
```

```
        write quotedate(count)
        write mainjobdesc(count)
        write eststock(count)
        write qnumofdays(count)
        write matprices(count)
        write totalprice(count)

End for
Close QuotesFile
End proc
```

Add Quote:

```
Proc AddQuote
unique = 0, uniquecust, presm, presnum, date, mile
Do proc ReadBackQuoteFile
While unique = 0
        output “Enter new quote reference: ”
        input quoteref [nqi]
        unique = Do Proc UniqueVal(quoteref[nqi])
Endwhile
While uniquecust = 0
        output “Enter customer reference: “
        input custno[nqi]
        uniquecust = Do proc FindCustomer(custno[nqi])
Endwhile
While date = 0
        Output “Enter the date the quote was produced: “
        Input quotedate[nqi]
```

```
        Date = Do Proc DateVal(quotedate[nqi])
End while
While presm = 0
        output “Enter description of the job”
        input mainjobdesc[nqi]
        presm = Do Proc PresenceCk(mainjobdesc[nqi])
Endwhile
While presnum = 0
        output “Enter the number of days you will spend on the job”
        input qnumofdays[nqi]
        presnum = Do Proc PresenceCk(qnumofdays[nqi])
Endwhile
While mile = 0
        output “Enter travel costs: “
        input mileage[nqi]
        mile = Do Proc RangeCk(mileage[nqi])
Endwhile
Stockcost = Do proc StockCalc
Labour = qnumofdays * 150
Pricenovat = total + labour+ mileage
VAT = 0.8*pricenovat
Totalcost = total + labour+ mileage + VAT
input totalcost[nqi]
output “Break down:”
output “                Material Prices: “,”£” total
output “                Labour Prices: “,”£” labour
output “                Mileage: “, “£”mileage
output “                VAT: “,”£” VAT
output “               Total price:”, “£”totalcost
```

```
ReWriteQuotesFile
End proc

Proc StockCalc
Do Proc ReadBackLinksFile
output “How many items of stock are required”
input numofitems
For add = 1 to numofitems
        output “ Enter stock ID: “
        input stockno
        Cost = Do proc FindStock (stockno)
        Total = cost + total
        Input total[nqi]
End for
Return total
End proc


Proc FindStock(stockref)
Do proc ReadBackStockFile
For find = 1 to nsti
        If (find[nsti] = stockref)
        cost = stockprice[nsti]
        End if
End for
Return cost
End proc
```

Delete a quote:

```
Proc DeleteQuote
Do proc ReadBackQuotesFile
output “Enter the quotereference: “
input quoteref[nqi]
for find = 1 to nqi
        if (find[nqi] = quoteref)
                Output “Customer reference: “, custno[find]
        Do proc LocateCustomer(custno)
                Output “Job Description”, mainjobdesc[find]
                Output “Is this the correct quote?”
                Input result
                If(result = ‘0’)
                        for del = find to nqi
                                quoteref[del] = quoteref[del+1]
                                custno[del] = custno[del+1]
                                quotedate[del] = quotedate[del+1]
                                mainjobdesc[del] = mainjobdesc[del+1]
                                qnumofdays[del] = qnumofdays[del+1]
                                mileage[del] = mileage[del+1]
                                totalcost[del] = totalcost [del+1]
                        End  for
                End if
        End if
End for
nqi = nqi –1
Do proc ReWriteQuotesFile
End proc
```

Change total price of quote:

```
Proc ChangePrice

Do Proc ReadBackQuotesFile

output “Enter quote reference: “
input quoteref[nqi]

For find = 1 to nqi

        If(find[nqi] = quoteref)

                Output “Customer reference: “, custno[find]
        Do proc LocateCustomer(custno)
                Output “Job Description”, mainjobdesc[find]
                Output “Is this the correct quote?”
                Input result
                If(result = ‘0’)
                        while pres = 0 and range = 0

                                output “Enter updated price: “

                                input totalcost[nqi]

                                pres = Do Proc PresenceCk(totalcost(nqi))

                                range = Do Proc RangeCk(totalcost(nqi))

                        end while

                End if

        End if

        Else

        output “Quote reference does not exist on the system.”

End for

Do proc ReWriteQuotesFile

End proc
```

Change number of days on the quote:

```
Proc ChangeDays

Do Proc ReadBackQuotesFile
```

```
output “Enter quote reference: “
input quoteref[nqi]

For find = 1 to nqi

        If(find[nqi] = quoteref)

                Output “Customer reference: “, custno[find]
                Do proc LocateCustomer(custno)
                Output “Job Description”, mainjobdesc[find]
                Output “Is this the correct quote?”
                Input result
                If(result = ‘0’)
                        while range = 0

                                output “Enter number of days expected to work: “

                                input qnumofdays[nqi]

                                range = Do Proc RangeCk(qnumofdays(nqi))

                        end while

                End if

        End if

        Else

                output “Quote reference does not exist on the system.”

End for

Do proc ReWriteQuotesFile

End proc
```

View quote by quote reference:

```
Prov ViewQuoteRef

output “Enter quote reference: “

Input quoteref[nqi]

Do proc ReadBackQuotesFile

For find = 1 to nqi

        If (find[nqi] = quoteref)

                output “ Customer reference: “, custno[nqi]
```

```
            Do Proc LocateCust(custno)
            output "Job description: " , mainjobdesc[nqi]
            output "Number of days to be worked: " , qnumofdays[nqi]
            output " Total cost: " ,"£" totalcost[nqi]
        End if
        Else
            Output "Quote reference does not exist."
End for
End proc
```

View Quote by customer reference:

```
Prov ViewQuotebyCustRef
output "Enter customer reference: "
input custno[nqi]
Do proc ReadBackQuotesFile
For find = 1 to nqi
        If (find[nqi] = custno)
        Do proc LocateCustomer(custno)
            output " Quote reference: ", quoteref[nqi]
            output "Job description: " , mainjobdesc[nqi]
            output "Number of days to be worked: " , qnumofdays[nqi]
            output " Price Breakdown: "
            output "              Material Prices: ","£" total
            output "              Labour Prices: ", "£"qnumofdays*150
            output "              Mileage: ", "£"mileage
            output "              VAT: ", "£"0.8*(total+(qnumofdays*150)+mileage)
            output "             Total price:", "£",totalcost
        End if
```

```
        Else
                Output “Quote reference does not exist.”
End for
End proc
```

Locate Customer:

```
Proc LocateCustomer(custno)
Locate(custref)
If (flag!=0)
        output “Customer name”, a_cust.fnamecust , a_cust.lnamecust
        output “Telephone number: ”, a_cust.telnocust
End if
Return custno
End proc
```

Serial File – Quotes

ReadBackStaffFile:

```
Proc ReadBackQuotesFile
Open QuotesFile
Read nqi
For count = 1 to nqi
        read quoteref(count)
        read custno(count)
        read quotedate(count)
        read mainjobdesc(count)
        read eststock(count)
        read qnumofdays(count)
        read matprices(count)
        read totalprice(count)
End for
Close QuotesFile
End proc
```

ReWriteQuotesFile:

```
Proc ReWriteQuotesFile
Open QuotesFile
Write nqi
For count = 1 to nqi
        write quoteref(count)
        write custno(count)
```

```
        write quotedate(count)
        write mainjobdesc(count)
        write eststock(count)
        write qnumofdays(count)
        write matprices(count)
        write totalprice(count)

End for
Close QuotesFile
End proc
```

Add Quote:

```
Proc AddQuote
unique = 0, uniquecust, presm, presnum, date, mile
Do proc ReadBackQuoteFile
While unique = 0
        output “Enter new quote reference: ”
        input quoteref [nqi]
        unique = Do Proc UniqueVal(quoteref[nqi])
Endwhile
While uniquecust = 0
        output “Enter customer reference: “
        input custno[nqi]
        uniquecust = Do proc FindCustomer(custno[nqi])
Endwhile
While date = 0
        Output “Enter the date the quote was produced: “
        Input quotedate[nqi]
```

```
        Date = Do Proc DateVal(quotedate[nqi])
End while
While presm = 0
        output “Enter description of the job”
        input mainjobdesc[nqi]
        presm = Do Proc PresenceCk(mainjobdesc[nqi])
Endwhile
While presnum = 0
        output “Enter the number of days you will spend on the job”
        input qnumofdays[nqi]
        presnum = Do Proc PresenceCk(qnumofdays[nqi])
Endwhile
While mile = 0
        output “Enter travel costs: “
        input mileage[nqi]
        mile = Do Proc RangeCk(mileage[nqi])
Endwhile
Stockcost = Do proc StockCalc
Labour = qnumofdays * 150
Pricenovat = total + labour+ mileage
VAT = 0.8*pricenovat
Totalcost = total + labour+ mileage + VAT
input totalcost[nqi]
output “Break down:”
output “                Material Prices: “,”£” total
output “                Labour Prices: “,”£” labour
output “                Mileage: “, “£”mileage
output “                VAT: “,”£” VAT
output “               Total price:”, “£”totalcost
```

```
ReWriteQuotesFile
End proc

Proc StockCalc
Do Proc ReadBackLinksFile
output “How many items of stock are required”
input numofitems
For add = 1 to numofitems
        output “ Enter stock ID: “
        input stockno
        Cost = Do proc FindStock (stockno)
        Total = cost + total
        Input total[nqi]
End for
Return total
End proc


Proc FindStock(stockref)
Do proc ReadBackStockFile
For find = 1 to nsti
        If (find[nsti] = stockref)
        cost = stockprice[nsti]
        End if
End for
Return cost
End proc
```

Delete a quote:

```
Proc DeleteQuote
Do proc ReadBackQuotesFile
output “Enter the quotereference: “
input quoteref[nqi]
for find = 1 to nqi
        if (find[nqi] = quoteref)
                Output “Customer reference: “, custno[find]
        Do proc LocateCustomer(custno)
                Output “Job Description”, mainjobdesc[find]
                Output “Is this the correct quote?”
                Input result
                If(result = ‘0’)
                        for del = find to nqi
                                quoteref[del] = quoteref[del+1]
                                custno[del] = custno[del+1]
                                quotedate[del] = quotedate[del+1]
                                mainjobdesc[del] = mainjobdesc[del+1]
                                qnumofdays[del] = qnumofdays[del+1]
                                mileage[del] = mileage[del+1]
                                totalcost[del] = totalcost [del+1]
                        End  for
                End if
        End if
End for
nqi = nqi –1
Do proc ReWriteQuotesFile
End proc
```

Change total price of quote:

```
Proc ChangePrice

Do Proc ReadBackQuotesFile

output “Enter quote reference: “
input quoteref[nqi]

For find = 1 to nqi

        If(find[nqi] = quoteref)

                Output “Customer reference: “, custno[find]
        Do proc LocateCustomer(custno)
                Output “Job Description”, mainjobdesc[find]
                Output “Is this the correct quote?”
                Input result
                If(result = ‘0’)
                        while pres = 0 and range = 0

                                output “Enter updated price: “

                                input totalcost[nqi]

                                pres = Do Proc PresenceCk(totalcost(nqi))

                                range = Do Proc RangeCk(totalcost(nqi))

                        end while

                End if

        End if

        Else

        output “Quote reference does not exist on the system.”

End for

Do proc ReWriteQuotesFile

End proc
```

Change number of days on the quote:

```
Proc ChangeDays

Do Proc ReadBackQuotesFile
```

```
output “Enter quote reference: “
input quoteref[nqi]

For find = 1 to nqi

        If(find[nqi] = quoteref)

                Output “Customer reference: “, custno[find]
                Do proc LocateCustomer(custno)
                Output “Job Description”, mainjobdesc[find]
                Output “Is this the correct quote?”
                Input result
                If(result = ‘0’)
                        while range = 0

                                output “Enter number of days expected to work: “

                                input qnumofdays[nqi]

                                range = Do Proc RangeCk(qnumofdays(nqi))

                        end while

                End if

        End if

        Else

                output “Quote reference does not exist on the system.”

End for

Do proc ReWriteQuotesFile

End proc
```

View quote by quote reference:

```
Prov ViewQuoteRef

output “Enter quote reference: “

Input quoteref[nqi]

Do proc ReadBackQuotesFile

For find = 1 to nqi

        If (find[nqi] = quoteref)

                output “ Customer reference: “, custno[nqi]
```

```
            Do Proc LocateCust(custno)
            output "Job description: " , mainjobdesc[nqi]
            output "Number of days to be worked: " , qnumofdays[nqi]
            output " Total cost: " ,"£" totalcost[nqi]
        End if
        Else
            Output "Quote reference does not exist."
End for
End proc
```

View Quote by customer reference:

```
Prov ViewQuotebyCustRef
output "Enter customer reference: "
input custno[nqi]
Do proc ReadBackQuotesFile
For find = 1 to nqi
        If (find[nqi] = custno)
        Do proc LocateCustomer(custno)
            output " Quote reference: ", quoteref[nqi]
            output "Job description: " , mainjobdesc[nqi]
            output "Number of days to be worked: " , qnumofdays[nqi]
            output " Price Breakdown: "
            output "              Material Prices: ","£" total
            output "              Labour Prices: ", "£"qnumofdays*150
            output "              Mileage: ", "£"mileage
            output "              VAT: ", "£"0.8*(total+(qnumofdays*150)+mileage)
            output "             Total price:", "£",totalcost
        End if
```

```
        Else
                Output “Quote reference does not exist.”
End for
End proc
```

Locate Customer:

```
Proc LocateCustomer(custno)
Locate(custref)
If (flag!=0)
        output “Customer name”, a_cust.fnamecust , a_cust.lnamecust
        output “Telephone number: ”, a_cust.telnocust
End if
Return custno
End proc
```

Serial File – Staff

ReadBackStaffFile:

```
Proc ReadBackStaffFile
Open StaffFile
Read nsi
For count = 1 to nsi
        read staffref (count)
        read fnamestaff (count)
        read lnamestaff (count)
        read oneadstaff (count)
        read twoadstaff (count)
        read threeadstaff(count)
        read pcodestaff (count)
        read telnostaff (count)
        read emtel (count)
        read ninum (count)
        read username(count)
        read password(count)
        read loa(count)
End for
Close StaffFile
End proc
```

ReWriteStaffFile:

```
Proc ReWriteStaffFile
Open StaffFile
Write nsi
```

```
For count = 1 to nsi
        write staffref (count)
        write fnamestaff (count)
        write lnamestaff (count)
        write oneadstaff (count)
        write twoadstaff (count)
        write threeadstaff(count)
        write pcodestaff (count)
        write telnostaff (count)
        write emtel (count)
        write ninum (count)
        write username(count)
        write password(count)
        write loa(count)
End for
Close StaffFile
End proc
```

Add Staff:

```
Proc AddStaff
unique = 0, presf = 0, presl = 0, presaone = 0, presatwo =0, presathree =0, pcode = 0, telnum=0,
emtel-0, nino =0 , uniqueuser=0, uniquepass=0, lev=0
Do Proc ReadBackStaffFile

While unique = 0
        output “Enter new staff reference: ”
        input staffref [nsi]
        unique = Do Proc UniqueVal(staffref [nsi])
```

```
Endwhile


While presf = 0
        output “Enter first name: “
        input fnamestaff [nsi]
        presf = Do Proc PresenceCk(fnamestaff [nsi])
Endwhile
While presl = 0
        output “Enter last name: “
        input lnamestaff [nsi]
        presl = Do Proc PresenceCk(lnamestaff [nsi])
Endwhile
While presaone = 0
        output “Enter address line 1: “
        input oneadstaff [nsi]
        presa1 = Do Proc PresenceCk(oneadstaff [nsi])
Endwhile
While presatwo = 0
        output “Enter address line 2 “
        input twoadstaff [nsi]
        presatwo = Do Proc PresenceCk(twoadstaff [nsi])
Endwhile
While presathree = 0
        output “Enter address line 3: “
        input 3adstaff [nsi]
        presathree = Do Proc PresenceCk(threeadstaff [nsi])
Endwhile
While pcode = 0
        output “Enter postcode: “
        input pcodestaff [nsi]
        pcode = Do Proc PostcodeVal(pcodestaff [nsi])
```

```
Endwhile
While telnum = 0
        output “Enter mobile number: “
        input telnostaff [nsi]
        telnum = Do Proc TelFormatCk(telnostaff [nsi])
Endwhile
While emtel = 0
        output “Enter emergency mobile number: “
        input emtel [nsi]
        emtel = Do Proc TelFormatCk(emtel [nsi])
Endwhile

While nino = 0
        output “Enter national insurance number: “
        input ninum [nsi]
        nino = Do Proc NIVal(ninum [nsi])
Endwhile
While uniqueuser=0
        output “Enter username: “
        input username [nsi]
        username = Do Proc UniqueVal(username [nsi])
End while
While unqiuepass =0
        output “ Enter password (At least 8 characters with at least one piece of punctuation or
        number) : ”
        input password [nsi]
        password = Do Proc PasswordCk(password [nsi])
End while
While lev = 0
        output “Enter level of access: “
        input loa [nsi]
```

```
        lev = Do Proc RangeCk(loa [nsi])
End while
nsi = nsi + 1
Do Proc ReWriteStaffFile
End proc
```

Delete Staff:

```
Proc DeleteStaff
Do Proc ReadBackStaffFile
output “Enter the staff reference: “
input staffref[nsi]
for find = 1 to nsi
        if (find[nsi] = staffref)
        Output"Full name: “, fnamestaff[find], lnamestaff[find]
        Output”Is this the correct staff member”
        Input result
                If(result = ‘0’)
                        for del = find to nsi
                                staffref[del] = staffref[del+1]
                                fnamestaff[del] = fnamestaff[del+1]
                                lnamestaff[del] = lnamestaff[del+1]
                                oneadstaff[del] = oneadstaff[del+1]
                                twoadstaff[del] = twoadstaff[del+1]
                                threeadstaff[del] = threeadstaff[del+1]
                                pcodestaff[del] = pcodestaff[del+1]
                                telnumstaff[del] = telnumstaff[del+1]
                                emtel[del] = emtel[del+1]
                                ninum[del] = ninum[del+1]
                                username[del] = username[del+1]
                                password[del] = password[del+1]
```

```
                                    loa[del] = loa[del+1]
                            End for
                    End if
            Endif
Endfor
nsi = nsi –1
Do proc ReWriteStaffFile
End proc
```

Change Staff Telephone Number:

```
Proc STelnum
telnum = 0
Do proc ReadBackStaffFile
output “Enter the staff reference: “
input staffref[nsi]
For find = 1 to nsi
        If( find[nsi] = staffref)
        Output"Full name: “, fnamestaff[find], lnamestaff[find]
        Output”Is this the correct staff member”
        Input result
                If(result = ‘0’)
                        While telnum = 0
```

```
                output “Enter mobile number: “
                input telnostaff [nsi]
                telnum = Do Proc TelFormatCk(telnostaff [nsi])
            Endwhile
        End if
    End if
End for
Do proc ReWriteStaffFile
End proc
```

Change Staff Home Address:

```
Proc SHomeAdd
presaone = 0, presatwo =0, presathree =0, pcode = 0
Do proc ReadBackStaffFile
output “Enter the staff reference: “
input staffref[nsi]
For find = 1 to nsi
    If( find[nsi] = staffref)
    Output"Full name: “, fnamestaff[find], lnamestaff[find]
    Output”Is this the correct staff member”
    Input result
        If(result = ‘0’)
            While presaone = 0
                output “Enter address line 1: “
                input oneadstaff [nsi]
                presaone = Do Proc PresenceCk(oneadstaff [nsi])
            Endwhile
            While presatwo = 0
                output “Enter address line 2: “
                input twoadstaff [nsi]
                presatwo = Do Proc PresenceCk(twoadstaff [nsi])
```

```
                    Endwhile
                    While presathree = 0
                        output “Enter address line 3: “
                        input threeadstaff [nsi]
                        presathree = Do Proc PresenceCk(threeadstaff [nsi])
                    Endwhile
                    While pcode = 0
                        output “Enter postcode: “
                        input pcodestaff [nsi]
                        pcode = Do Proc PostcodeVal(pcodestaff [nsi])
                    Endwhile
                End if
            End if
End for
Do proc ReWriteStaffFile
End proc
```

<u>View staff by staff reference:</u>

```
Proc ViewStaffRef
output “Enter staff reference: “
input staffref[nsi]
Do proc ReadBackStaffFile
For find = 1 to nsi
        If (find[nsi] = staffref)
                output “First name: “, fnamestaff[nsi]
                output “Last name: “, lnamestaff[nsi]
                output “Address line 1: “, oneadstaff[nsi]
                output “Address line 2: “, twoadstaff[nsi]
                output “Address line 3: “, threeadstaff[nsi]
                output “Postcode: “, pcodestaff[nsi]
                output “Telephone number: “, telnostaff[nsi]
```

```
        output "Emegency number: ", emtel[nsi]
        output “National Insurance Number: “, ninum[nsi]
End if
Else
        Output “Staff reference does not exist.”
End proc
```