1. Introduction

- The language used is at the discretion of the centre.
- The finished work should purport to do something useful.
- Keep the volume of work to a minimum.
- Keep things as simple as possible.
- Real end user.
- Demonstrate the 8 programming techniques listed in the syllabus.
- Satisfying to the student.
- 36 out of the 50 available marks are for producing and testing the solution.

2. Example End User

- As real as possible.
- Keep a stock of them for any student who cannot find one.
- Will be involved at different stages so should be available.

3. Intention of this example project

These notes take the form of a project that a student might hand in. There are obvious issues with the language used in any such exemplar piece of work, and for that reason the programming has been largely omitted, though comments are made.

It should be accepted that this is not intended as a perfect piece of work but it is meant to reflect the sort of quality that a good candidate will hand in.

My example project is completed in 8 sides. If additional volume is considered in order to accommodate the code, the test screens and the designs of the screens, it will still be less than 20 sides of A4. This is a reasonable target volume for students. Some will want to do a far more grandiose piece of work and they will not be penalised for doing so, but it is not necessary and whether or not the added educational value is worth the extra time that the student will have to spend on the work is a question that only the teacher can answer.

4. A sensible scheme of work

The main reason for the change in practical work was to relieve some of the pressure on students because of the large volume of work which was expected. Centres should not expect too much from their candidates otherwise the whole point about the change will be nullified.

It is expected that the work (with the exception of finding out about the problem from the end user) will occupy practical lessons for about 4 weeks. In other words about 8 hours work. If it takes much longer than that then the student is doing more than expected.

Note that the work is angled very much toward the production of a working solution, with a relatively small amount of documentation expected. The finished product should reflect this.

The various techniques will need to be taught before work begins on the actual project so that students can do most of the work without help from the teacher. The way that the work is taught or the resources used to do the teaching will depend on the centre and upon the language used. If a centre is going to use VB.NET (or even earlier versions) an excellent resource is “Computing projects in Visual Basic.NET” by Derek Christopher and published by Payne-Gallway Ltd.
Centres should consider that a suitable time plan for the work over the academic year is to teach the techniques for the project over the first term of the year. The students than complete their individual projects, largely during lesson time, and then the second part of the year is spent teaching the software packages and techniques that students will need for their major project in the second year.

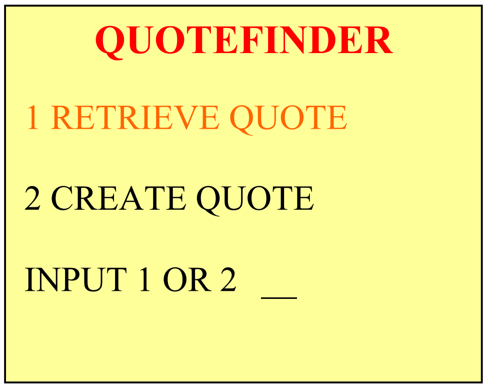
a) Problem / Task identification:

The end user for my project will be my uncle, Mr. Imzamam al Haq. He has a decorating business and needs to solve a problem that he has when he provides estimates for people.

There are a number of differently priced finishes which he finds difficult to remember. The price of the decorating of a room depends on the size of the room and the finish chosen. He would like to be able to input the dimensions and for the prices to be displayed on the screen.
He would like this to be on a computer rather than on paper so that the agreed quotes can be stored on a computer file.

b) Program Design.

Screens:

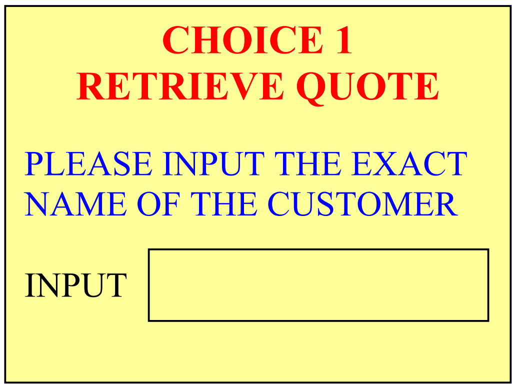


QUOTEFINDER

1 RETRIEVE QUOTE

2 CREATE QUOTE

INPUT 1 OR 2 __



CHOICE 1
RETRIEVE QUOTE

PLEASE INPUT THE EXACT NAME OF THE CUSTOMER

INPUT

INPUT SCREEN 1 (NOTE DUAL NATURE OF BOTH SCREENS) OUTPUT SCREEN 1a

File Structures:

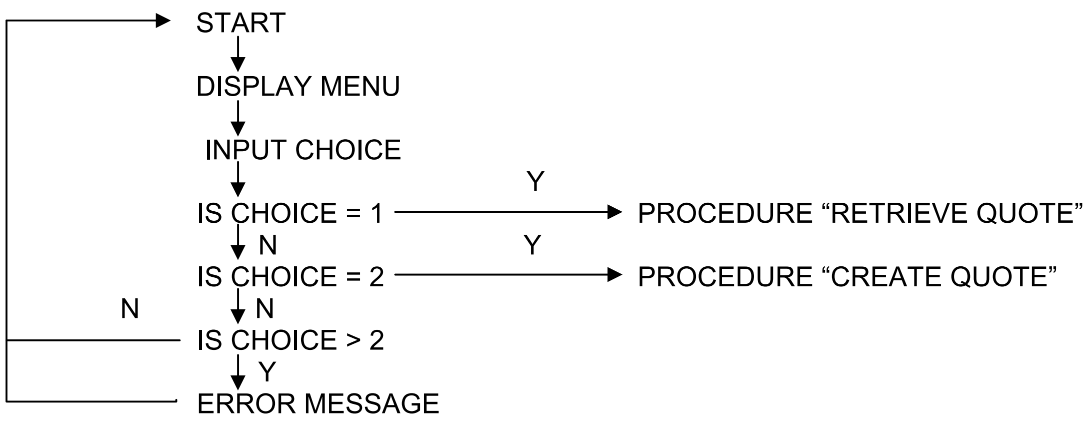
File 1 is the file of finishes

Finish (String)	Price (Currency)
e.g. Emulsion paint	.11

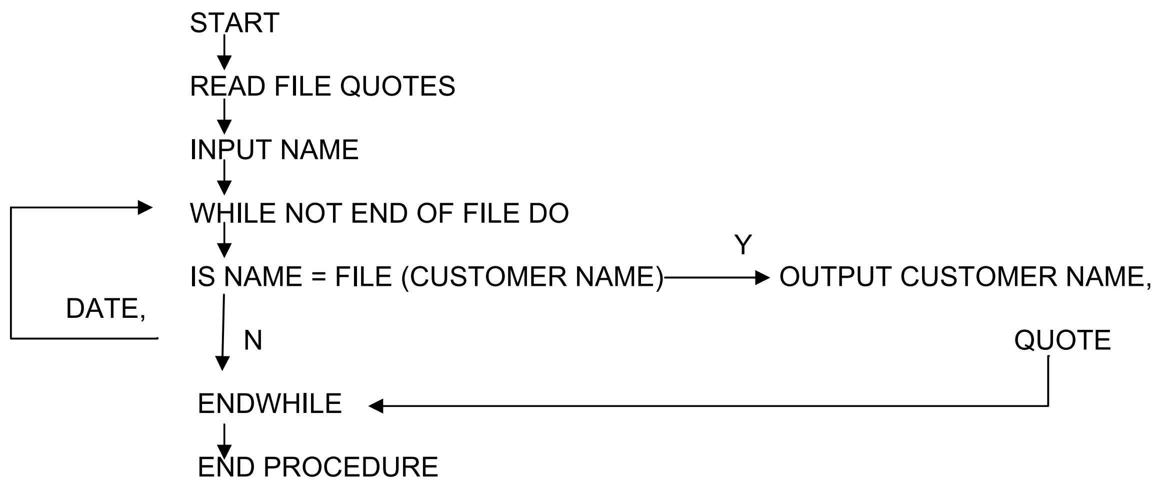
File 2 is the file of quotations

Customer name	Quotation date	Quote
e.g. Mr. W. Akram	05/08/05	123.45

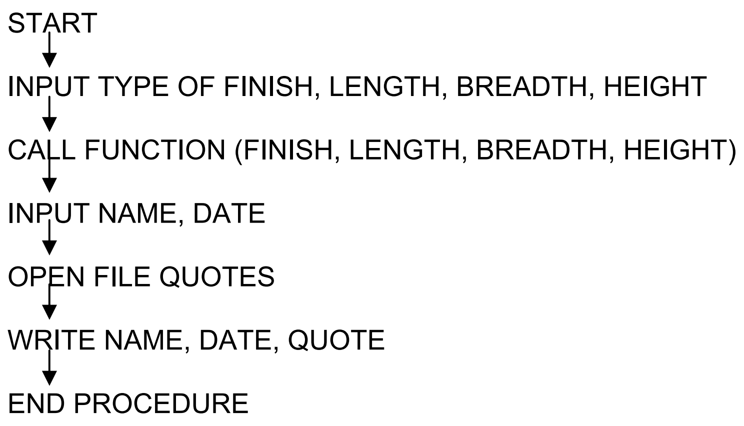
Main Algorithm



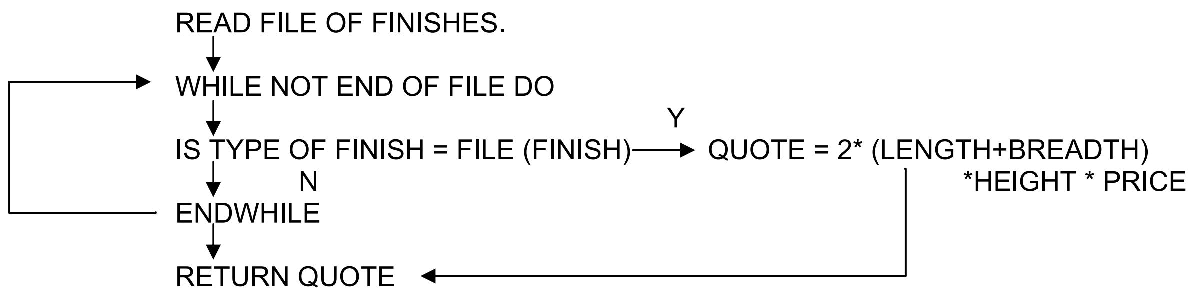
PROCEDURE “RETRIEVE QUOTE”



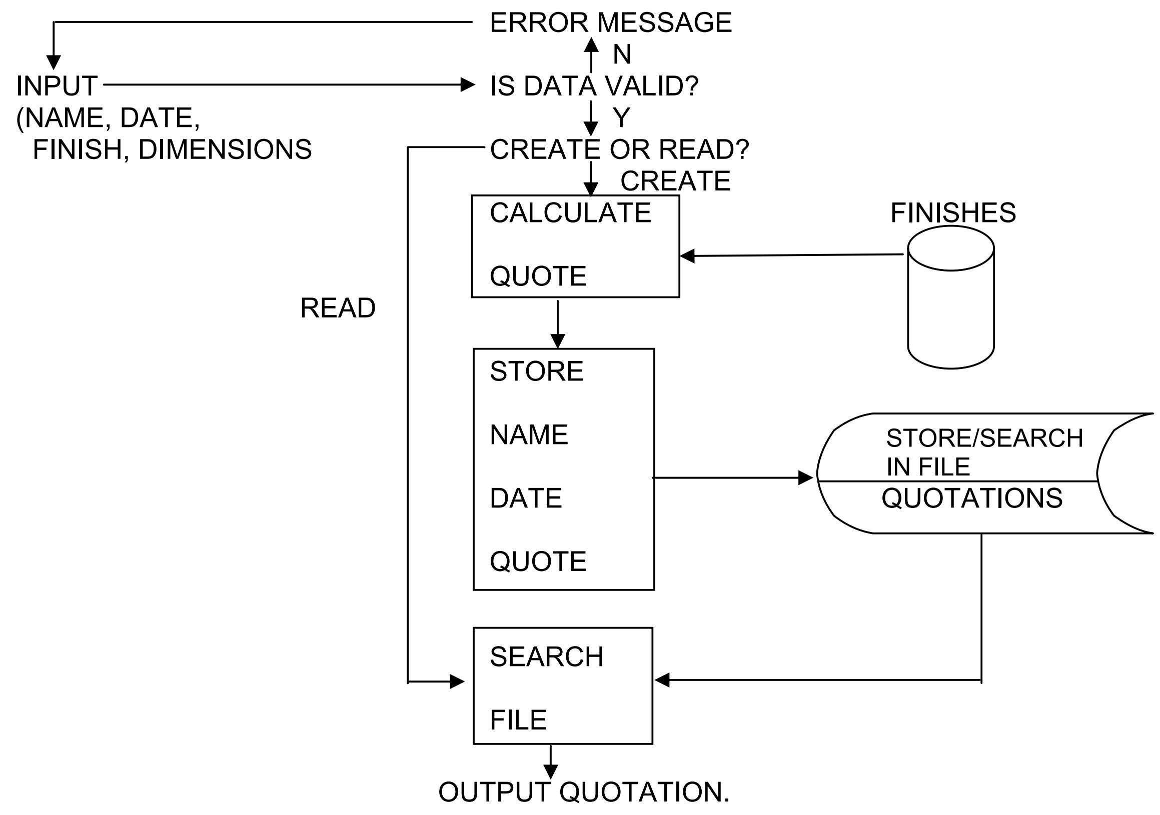
PRODEDURE “CREATE QUOTE”


START
INPUT TYPE OF FINISH, LENGTH, BREADTH, HEIGHT
CALL FUNCTION (FINISH, LENGTH, BREADTH, HEIGHT)
INPUT NAME, DATE
OPEN FILE QUOTES
WRITE NAME, DATE, QUOTE
END PROCEDURE

FUNCTION (FINISH, LENGTH, BREADTH, HEIGHT)



DATA FLOW



Validation of the input data is not included here so I would only earn 4 or 5 out of 6, but it would be easy to include simple range and datatype checks on input.

c) (i) IMPLEMENTING THE PROGRAM:

This will be a simple listing of the program code which produces the desired outcomes. This will obviously differ according to the language used, but a problem like the one here should not take more than about 2 sides of code.
Ensure that the code is fully annotated. A good guideline for annotation is the set of flow charts / pseudocode produced in (b). Each separate line/instruction/box should be annotated in the code. This will give enough for the code to be followed while not doing too much.

(ii) GOOD PROGRAMMING STYLE

- Header identifying program/version/author/date/school/language
- Plenty of remark statements (as in (i))
- Use of sensible names
- Use of procedures/functions
- Use of indentation

(iii) PROGRAMMING SKILLS.

The candidate should simply highlight a relevant piece of their code that illustrates each of the eight skills and write one or two sentences about each one.

RECORDS	Each of my two files contain records about specific entities. e.g. the quotations file contains the record Mr. W. Akram 05/08/05 123.45
DIFFERENT DATA TYPES	I have used a string data type for customer name, real for the price and the quote integer for the length, breadth and height.
SELECTION	My main menu offers a selection of two options leading to different procedures.
ITERATION	In my procedure "Retrieve Quote" I use a WHILE……ENDWHILE structure which reads through the file sequentially, one record at a time, this is evidence of iteration.
PROCEDURES	My solution uses two sub programs, or procedures, "Retrieve Quote" and "Create Quotation".
FUNCTIONS	I have used a function called FUNCTION which has parameters FINISH, LENGTH, BREADTH, HEIGHT, and returns a single value QUOTE.
SEARCHING	In each of my procedures I have used a WHILE….ENDWHILE structure to search for specific values.
FILES	I have used two files. QUOTES is dynamic and can be added to. FINISHES is static and the contents are essentially a look-up table.

(NOTE: Code needs to be clearly annotated to show each of these features).

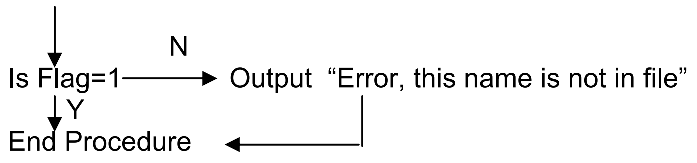
d) TESTING

TEST PLAN

INPUT	**PURPOSE**	**EXPECTED RESULT**	**ACTUAL RESULT**	**EXPLANATION**
1 CHOICE = 2	Does this lead procedure “Create Quote”	Asks for FINISH	Asks for FINISH	
2 CHOICE=3	Should not go to procedures.	Produces error Message	Outputs “Please enter 1 or 2”	
3 CHOICE=1 AND NAME = W.AKRAM	To test how accurate names need to be	Mr. Akram’s quote	No output	“Mr.” not input
4 CHOICE=1 AND NAME = MR.W.AKRAM	To test search process	Mr. W.Akram’s quote	Mr. W. Akram 05/08/05 123.45	
5 CHOICE=1 AND NAME = SHOAIB	What happens if not on file	Message to say not on file	No Output	Need to include error message
6 CHOICE=2 F=EMULSION L=200 B=200	To check output	17600	No Output	No command after function

At this point the algorithm and code were altered.

In “RETRIEVE QUOTE” a flag was inserted. The flag was initialised to 0 before the loop and was altered to a 1 if the Name = File (Customer name) Between Endwhile and End Procedure the following was inserted



Test :

7 INPUT: CHOICE=1
AND NAME=SHOAIB

OUTPUT: Error this name
is not in file.

Note: The improvement has worked.

The second change is to insert a new command: “Output Quote” in the “Create Quote” procedure before “Input Name, Date.”

Test			

8	Input	CHOICE=2
		F=EMULSION
		L=200,B=200
		H=200
			OUTPUT: 17600

Note: The improvement has worked. (Though if I were my teacher I would want an explanation of the weird units being used!)

Note: Not all parts of a sensible solution have been tested. I am missing too many validation routines for full marks. However, I have tested MY solution and I have done 8 tests.
Impressively, I have also found errors in my solution and have corrected them and proved it in further tests. This must be worth 6 or 7 out of 8.

Note: An additional 4 sides necessary to show the eight test screens.

e) (i) Technical Documentation.

Data used:

CHOICE:	Integer, can take values 1 or 2, used to select operation to be carried out.
NAME:	String, variable length, used to allow user to state which quote is required.
DATE:	String, fixed length of 8 characters, used to identify date of quote.
QUOTE:	Real, used to store value of quotation.
FLAG:	Boolean, used to decide whether a quote has been found or not.
FINISH:	String, used to allow user to select the finish required.
PRICE:	Real, the cost per square centimetre.

LENGTH, BREADTH, HEIGHT: Integer in centimetres, to enable the dimensions of the room to be input.

Back-up: The quotation file will be small and therefore can be stored on a floppy disk. The file should be copied to the disk each evening after all visits have been carried out for the day. The disk is then stored away from the computer.

Data flow: See diagram on page 5.

Algorithms: See flow diagrams on pages 3 and 4

Program listings: See listings on pages ?

Hardware and software:

The software has been created by writing a high level language program in ? language.
The program will run on any machine with a ? translator.
Hardware: Processor (speed not important as there are no graphics)
Screen/Keyboard (to act as user interface)
Hard disk (to store program code and files)
Floppy disk drive (to allow backup to be taken)

(ii) Installation:
The program is called Quotefinder,
Place the floppy disk in the drive and copy Quotefinder onto the hard disk.
It is also necessary to copy the static file Finishes and the file Quotes on to the hard disk.
To run the program simply type in Quotefinder at the C> prompt.