Software design fundamentals Level 3

Notes

for

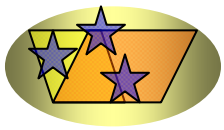
City & Guilds

7540 Unit 033

©Tench Computing Ltd

Tench Computing Ltd
Pines
Glendale Road
Burgess Hill
West Sussex
RH15 0EJ

Web address: www.tenchcomputing.co.uk
Email address: jtench@globalnet.co.uk



About the author: *Jackie Tench MSc, ACIB, Cert Ed*

Jackie started her working career in branch banking with the Midland Bank (now HSBC) and was transferred to their Computing Department after achieving 100% in their ability test for programmers. She then worked for more than a decade in this department and was one of the first women to achieve a junior management grade at the age of 21. She attended a significant number of IBM programming training courses during her time there.

Jackie was the first woman to pass the ACIB (Associate Chartered Institute of Bankers) examinations in the Midland Bank (HSBC) and the youngest person at 21 years of age.

Jackie then left to raise a family but still found time to teach part-time at a college in Sheffield and to obtain a MSc in Computing and a Cert Ed in teaching.

When her children were old enough Jackie returned to work full-time and was a Senior Lecturer in Software Engineering and Computer Studies at a college in Brighton for nearly 10 years teaching all levels up to and including HND.

Therefore, Jackie has considerable business knowledge and qualifications plus wide experience in practical computing and training – covering areas such as structured design, analysis, coding, testing and implementing software applications plus training students to fulfil an important role in the computer industry.

Jackie has worked as a consultant for several blue chip companies and examination boards using her software engineering and educational training skills and is now one of the foremost experts in computing with an extensive knowledge of programming languages and applications.

Copyright ©1999 Tench Computing Ltd

Microsoft, Windows, Windows NT or other Microsoft products referenced herein are either the trademarks or registered trademarks of the Microsoft Corporation. Other trademarks for products referenced herein are also acknowledged.

All rights are reserved and no part of this training manual may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the purchase of a licence.

This training manual is sold subject to licence and on condition that it shall not, by way of trade or otherwise, be lent, re-sold, hired out or otherwise circulated without the prior consent of Tench Computing Ltd in any form of binding or cover other than that in which it is issued and without a similar condition being imposed on the subsequent purchaser. Any program listings within this training manual may be entered, stored and executed in a computer but they may not be reproduced for publication.

Contents Page

[This page is intentionally blank]

Files

Software uses files to store data that needs to be used at a later time possibly by another application. There are two main types of file access that can be used in software: sequential and random, but the simplest to understand is a sequential file. Files can be stored in various formats not all of which are readable e.g. binary. Text files are readable.

Sequential files

Sequential files can be stored and accessed on tape or disk. A sequential text file normally consists of records and each record is made up of related fields.

For example a product record may be made up of the fields ProductCode, Description, Quantity, Price.

Field name	**ProductCode**	**Description**	**Quantity**	**Price**
Data	AT09	Flat Screen 19”	109	265.99
	PF01	Monitor 17”	20	50.95
Field size maximum (in bytes)	4	30	6	10
Record size	50 bytes			

The data in the file would be held as shown below:

```
AT09Flat Screen 19”                    109    265.99EORPF01Monitor
17”                    20     50.95EOREOF
```

Each record is a fixed length, 50 bytes in the above example and one record follows another in the file. Each record normally has a delimiter character at the end shown as EOR in the example. There is also a delimiter character at the end of a file shown above as EOF. Sequential files can have variable length records.

Before a file can be used by software it has to be opened. The basic methods for accessing a sequential file are input, output or append. After use the software must close a file. Files output by software must be checked to ensure that the correct data has been written to them.

Field size

A field size must be as small as possible because a field is in every record in the file. For instance, if a field is required to represent a colour, the colour could be represented by a code rather by text to make the field size smaller. For instance the codes 1, 2 and 3 OR R, Y, M, could represent the colours Red, Yellow and Magenta respectively. This means that instead of the field having a maximum size of 7 (Magenta) it would have a size of 1. So a file with 3,000 records will save 3000 x 6 bytes, which is 18000 bytes.

Dates would also be held in a file using the smallest possible field size. This means that extra characters such as / or – are not added to dates in a file. Dates in a file would be held as ddmmyyyy e.g. 23042003. Extra characters are only added to make the dates more readable when they are displayed on screen or printed out.

Input file

A sequential file can be opened as an input file so that the software reads data in the file. When a file is opened as input data it is read into a memory storage area called a buffer ready for the software to access it. The amount of data read into memory depends on the size of the buffer area provided. A sequential file is read from the beginning one record at a time. So, if you need to access the 50th record you must first read the 49 records before it in the file. The file must be closed after use.

A file opened as input must exist. A programmer must include code to check that the file does exist before attempting to read from the file. If the file does not exist and the software does not check then the software will crash with a run-time error.

Output file

A file can be opened as an output file and data will be written to the file by the software. Data is stored in a temporary buffer in memory before being written to the file. A file opened as output will overwrite any file that already exists with the same name in the selected directory. Records are written to a sequential file one after the other. It is important that a file opened as output is closed as the close writes any remaining data from the memory buffer to the file and adds the end of file marker.

Append file

An append to a file is used to write data to a file but if the file already exists, with the specified name, the data will be written at the end of the existing file. If the file does not already exist then a new file will be created. The new records are written to the sequential file one after the other. The file must be closed after use.

Limitations of a sequential file

Slow access. A sequential file provides a slow method of access to data. For example if a file has 5000 records and only record numbers 1000, 2500 and 4580 need to be accessed, all the records up and including record number 4580 will need to be read.

Sorting required. Since records are written to a sequential file one after the other a sequential file normally has to be sorted into order after the initial records have been written. For instance a file that contains product records would be sorted into product number order.

Amendments. Records in a sequential file cannot be amended in place. So if an amendment is required to a customer address for record number 50, records 1-49 have to be written to a new output file, record number 50 amended and written to the output file and then all the remaining records written to the output file.

External filename

An external filename is the location and name of the file on a storage device e.g. **c:\customer\labels.fl**. An external filename is only used once in a program to assign it to an internal filename.

Internal filename

An internal filename is a variable name used for the file within a program. The external filename is assigned to the internal filename when the file is opened and then the internal filename is always used within the program to refer to the file. The external filename is assigned to an internal filename so that if the location of the file is changed only one amendment to the program is required, the external filename, because the internal filename will not change.

File layout

When designing a program the file layout for any files must be specified. The file layout specifies the access method, the external name, the internal name, the record name, the size of a record, the type of the record, the field names, field sizes and field data types.

For example a file for products could have a file layout as follows:

Products file layout

External filename	D:\mycompany\products.fl
Internal filename	ProductFile
Access method	Sequential
Size of record	50

Record description

Record name	Product
Type of record	Fixed length

Field name	**Field size**	**Data type**
ProductCode	4	String
Description	30	String
Quantity	6	Integer
Price	10	Floating point includes 2 decimal places

It is important to document the layout of a file because the file may be used in several programs. A programmer needs to know the details about a file before designing software to access it. It also means that consistent names are used for the internal filename, record name and field names in any software that accesses the file.

Random Access files

Random Access files are made up of fixed length records and can only be used on disk. Each record is made up of fields that store data.

Below is an example of a random access file with 3 fields per record. The first field is a 1-byte record number. The second field is a 5-byte string that holds the name of a person. The third field is a 2-byte string that holds the age of the person. The length of each record in this file is 8 bytes (1 + 5 + 2). The first sequence of 8 bytes belongs to the first record, the second sequence of 8 bytes to the second record and so on. Each record stores data about a specific person (i.e. the record number, name and age).

Record	1								2								3							
Data	1	J	O	H	N		2	0	2	J	A	N	E		5	5	3	R	O	N			4	3
Field	1	2					3		1	2					3		1	2					3	

Random access files allow any record in the file to be accessed without first having to read the records before the required record in the file. So in the example shown above to read record 3 you do not need to read records 1 and 2 first but can access it directly.

The way that records are read from or written to a random access file is dependent on the programming language used. In some languages you define a record by specifying its fields and field types and then access a record by giving its record number (position in the file) e.g. 3. In other programming languages a record is accessed by its address, which is the number of bytes from the beginning of the file. So for the record in position 3, in the above file, the number of bytes would be 16 as the first byte in the file is 0.

The formula for working out the address of a record is given below:

(record number – 1) * number of bytes in record

Random access files can be opened for input to read records from the file or output to write records to the file. Some programming languages allow files to be opened as input-output which means that records can be read from and written to the file. This allows a record to be read, updated and then written back to the file.

Advantages
Random access files provide fast access to a record in the file. If record number 2000 needs to be accessed it can be moved to directly and read without having to read records 1-1999.
Records can be read, amended and then written back to the same position in the file.

Disadvantages
Random access files must have fixed length records otherwise the address of a record cannot be worked out.
Random access files cannot be used on tape.
Unused records take up space in the file as shown below where some records have been deleted and are no longer accessible but the space they occupied is still held in the file.

Record 1	Record 2	Record 3	Record 4	Record 5
Data	Deleted	Data	Data	Deleted

Error conditions

Whenever an access is made to a file e.g. open, read, write, update, close, a status code is returned which indicates whether the access was successful or not and sometimes an error code is supplied as well to indicate what type of error has occurred. The status code should always be checked so that the software can take action if a file access was unsuccessful. An unsuccessful file access, which is not checked, and action taken in the software can cause a program to crash resulting in corruption of the data in a file.

The types of error that can occur when accessing a file are:

- files does not exist – this error can occur when a file is being opened.
- record does not exist –this error can occur when a record is being read.
- read past end of file – this error can occur when the software tries to read a record past the end of file and is normally because a read is done without checking for the end of file. This type of error should be found when testing the software and not when it is operational.
- hardware fault – this error can be found at any time when access is made to a file. The software should display a message to the user and close down so that the data in the file does not become corrupted by further accesses.
- File not open – this error occurs if access to a record is attempted before the file is opened.
- Attempt to output to a file opened for input – this error occurs if an incorrect access is attempted for a file for instance a write to a file that is INPUT only.
- Disk full – this error occurs if a write to a file is attempted and there is no more space available on the disk.
- Attempt to input from a file opened for output - this error occurs if an incorrect access is attempted for a file for instance a read from a file that is OUTPUT only.
- Attempt to open a file that is already open – this error occurs if an open is done for a file which is already open.
- Attempt to access a file with incorrect permission – this occurs when a file access is attempted which is not allowed e.g. trying to write data to a file that is READ only.

Some of these errors should be found when software is tested e.g. reading past the end of file, but some will occur when the software is operational. A programmer should always check the status codes returned after an access to a file and take appropriate action in the software e.g. display a message to the user or close a file.

Validation

It is important that software does not allow invalid data to be entered. If invalid data is accepted by software then it may crash with a run-time error. If numeric data was required by software because it was to be used in a calculation and non-numeric data was entered and accepted, when the software tried to do the calculation it would crash because the computer cannot do calculations on non-numeric data.

Software developers should write routines in the software that check the entered data to make sure it is valid. A tester must use test data that is invalid to prove that the software rejects invalid data.

Types of validation

Date checks

Whenever a date is used as input, the software should check that it is valid. The type of check made depends on the format of the date entered. If a date is entered in the format dd/mm/yyyy then full checks are required.

The days (dd) must be inside the boundaries for days in a month. For most months this is between 1 and 31 but for the months April, June, September and November the days are between 1 and 30. For February the days are between 1 and 28 except for a leap year when the days are between 1 and 29.

The month (mm) is checked to make sure it is between 1 and 12.

The year (yyyy) cannot be checked properly but could be checked for the first two digits being 19 or 20. It depends on the application what the range of years can be. If the software was for a museum which was holding details about historical artefacts then the range of years could not be checked because there would be a wide range of years allowable.

The format in which the date is entered should be specified in the design specification and it is up to a tester to decide what test data and tests should be done to make sure that the software will not accept invalid dates.

Range checks

Some data input may be specified as restricted to a range. For instance a value entered may be restricted to values between 100-500. A range check must be written in the software to prevent any other values being accepted. The range for data input should be specified in the design specification. A tester must decide what test data to use to prove that invalid values would not be accepted by the software.

For data input restricted to the values 100-500 the test data would be 99, 100, 101, 499, 500 and 501. This data tests the boundaries of the values. The values 99 and 501 should be rejected and the values 100, 101, 499 and 500 should be accepted.

Type checks

Numeric

Input data that has numeric calculations done on it must be checked to ensure that it is numeric. If non-numeric data is allowed as input the software will crash when the computer attempts to do the calculations. The software must reject any data input that is non-numeric. A tester must create test data that includes alphanumeric data to ensure that the software rejects the non-numeric data.

Alphabetic

Input data that must be alphabetic can be checked to ensure that it is alphabetic. Alphabetic data only includes the characters A-Z and a-z therefore data that includes spaces and special characters e.g. a hyphen (-) cannot be alphabetic.

Check digits

Bank account numbers, customer numbers, stock code numbers are all keyed into computer systems. If a number is keyed in incorrectly then the wrong records will be accessed. Check digits are added to these types of data input as a means of trapping the errors. The check digit is calculated by using the existing digits in the number. A widely used method for calculating check digits is modulus 11 which traps over 99% of errors. Barcodes normally incorporate a check digit. Some systems use modulus 10 check digits. The calculation for a modulus 10 check digit is the same as for modulus 11 except that 10 is used in place of 11 for the calculation and the check digit can only be in the range 0-9.

Create a check digit (modulus 11)

A check digit is created and then added as the last digit in a number.
To create the check digit, using modulus 11, a weight is assigned to each digit in the number. The rightmost digit in the number is assigned the weight 2, the digit at its left is assigned the weight 3, the next digit at the left the weight 4 and so on for however many digits are in the number.
Each digit is then multiplied by its weight.

Number	3	5	2	4
Weight	5	4	3	2
Weight multiplied by Number	15	20	6	8

The products are then added together 15 + 20 + 6 + 8 = 49

The result is then divided by the modulus (11) 49 /11 = 4 remainder 5

The remainder from the division is then subtracted from the modulus (11) 11-5 = 6

The result is the check digit and this is added as the last digit in the number to give:

35246

Note the following exceptions for modulus 11
If the remainder is 0 the check digit is 0
If the remainder is 10 the check digit is X

Check a check digit (modulus 11)
When data input is a number containing a check digit the software does a calculation on the digits in the number as follows.

A weight is assigned to each digit in the number. The rightmost digit in the number is assigned the weight 1, the digit at its left is assigned the weight 2, the next digit at the left the weight 3 and so on for however many digits are in the number.
Each digit is then multiplied by its weight.

Number	3	5	2	4	6
Weight	5	4	3	2	1
Weight multiplied by Number	15	20	6	8	6

The products are then added together 15 + 20 + 6 + 8 + 6 = 55

The result is then divided by the modulus (11) 55 /11 = 5 remainder 0

If the remainder is 0 then the number is a valid modulus 11 number.

If the number 35246 was keyed in incorrectly as 32546 and the modulus 11 check done as follows:

Number	3	2	5	4	6
Weight	5	4	3	2	1
Weight multiplied by Number	15	8	15	8	6

The products are then added together 15 + 8 + 15 + 8 + 6 = 52

The result is then divided by the modulus (11) 52 /11 = 4 remainder 8

The remainder is not 0 so the number is not a valid modulus 11 number. The software should reject this number as invalid.

If software requires input of a number with a check digit then a tester must create test data that includes valid numbers and invalid numbers to make sure that the invalid numbers are rejected by the software. This means you have to know how to create numbers with check digits to create the test data.

Presence check

Input data may be specified as required which means it cannot contain spaces. Fields such as customer account number for a sales order or student enrolment number for a student record must be present and cannot be omitted. The software must check that these fields do not contain spaces. A tester must create test data that includes spaces for this data to ensure that the software will reject the spaces.

Character count

Input data such as a customer account number may be specified as fixed length for example 6 characters. The software must check that data that has less or more characters is rejected. A tester must create test data that contains the correct number of characters and data that contains less and more characters to ensure only the correct length data is accepted.

Format check

Input data may be specified as being in a set format for example two letters followed by four digits (AG2372). The software must check that the data contains the correct format. A tester must create test data that contains the correct and incorrect format to ensure that data with the incorrect format is rejected.

Lookup

Input data may be specified so that it can only contain certain specified values. For example books may only have a set number of categories – Fiction, Historical, Information Technology. The software must check that only these categories can be entered. This can take the form of a list presented to a user so that they can only select from the displayed list.

Hash totals

Hash totals are used as controls for data. A hash total is a total of data that has no meaningful numerical value such as a total of customer account numbers. This type of total can be used to check that records have not been incorrectly inserted or deleted from files or data structures. (Control totals that have numerical values would a total of the number of items input or the total of customer balances.)

Error conditions

The following are examples of the type of error conditions that can occur in software. The software must be designed so that it checks for these errors. If possible the software should take action to prevent the error occurring. If some action must be taken by the user then an error message should be displayed on the screen.

Division by zero	This will always cause a program to crash. It is a designer's responsibility to ensure that a division by zero does not occur by checking the data used for division.
File does not exist	This will cause a program to crash. It is a designer's responsibility to ensure that a check is made that a file opened for input exists before access is made to it. If the file does not exist an error message should be displayed to the user.
Record does not exist	Reading past the end of a file can cause this. When reading a file, the EOF marker should always be looked for.
Data mismatch	This can be caused by a program trying to put the wrong type of data into a variable. Input data should always be checked to ensure it is the correct type.
Calculation error	This is caused by trying to do a calculation on non-numeric data. Input data should always be checked to ensure it is the correct type.
Array error	This is caused by using a subscript for an array that is outside the bounds of the array. This may not cause the program to crash but may make it produce incorrect results.

A run-time error is an error that causes a program to crash. A designer should design software so that it does not crash.

A logical error in software is an error that the programmer has made in the logic of the code. A logical error may cause a run-time error it or may just produce incorrect results as output.

Screen error messages

The error messages that are to appear on the screen when an error occurs in the software must be documented. A designer must specify the error messages that are to be produced by a program. Normally error messages are given an error code number so that when the user manual is created the error messages can be listed in number order so that they are easier to locate.

When a program contains validation routines then error messages must be displayed if a user enters incorrect data. The error message must be understandable by a user. For instance, if a validation routine checks that an account number is a valid modulus number and the user enters an incorrect number then the error message should display 'Incorrect account number entered' and not 'Invalid check digit'. This is because a user will not understand what a check digit is and will not know how to respond to the error message. Displaying 'Incorrect account number entered' means a user does not need to look up what the error message means but will re-enter the account number.

Example error messages

Error code	Error message
001	'File does not exist'
002	'Incorrect member number entered'
003	'Member name must be entered'
004	'Value must be M or F'

Validation checks

A designer must specify any validation checks that are to be made on the data input into a program. For a customer record the validation checks could be as follows:

Field	Validation
Customer account number	Check digit modulus 10 Must not be spaces Input terminated by 999999
Name	Must not be spaces
Balance	Must be numeric

The alphanumeric fields for Title, Name, Address and Postcode are not validated as any characters can be entered into these fields. The telephone number does not need to be validated because it is not used in a calculation and is entered as alphanumeric data.

Questions 2

1. Design a file layout for a randomly accessed member file named **member.dat** to be stored on the **c:** drive in the **club** directory. The file must contain the following member details: member number, name, address, telephone number, date of birth, membership type (Full, Social, Gym, Tennis or Squash), date joined, subscription due date and subscription amount.
2. If the date format to be entered into software is dd/mm/yyyy specify whether the following dates should be accepted or rejected:
    a. 31/09/2002
    b. 29/02/2003
    c. 31/08/2002
    d. 30/11/2003
    e. 31/03/2003
    f. 29/02/2000
    g. 30/04/1998
    h. 31/06/2002
3. A data entry must be numeric in the range 25 to 100 (inclusive). Specify which of the following values should be accepted or rejected.
    a. 25
    b. 99
    c. 24
    d. 100
    e. 26
    f. 101
    g. 66
4. A data entry must be 6 characters and a valid modulus 11 number. Specify which of the following values should be accepted or rejected.

    Note that X counts as a 10.

    a. 19723Y
    b. 921343
    c. 11458
    d. 66135X
    e. 125350
    f. 2960X
    g. 326410

5. A data entry must be 6 digits and a valid modulus 10 number. Specify which of the following values should be accepted or rejected.

    a. 232411

    b. 36789X

    c. 169430

    d. 661239

    e. 11789

    f. 529648

6. A data entry must be in the format of two alphabetic characters followed by 5 numeric digits. Specify which of the following values should be accepted or rejected.

    a. RG6398

    b. PL34512

    c. HBS8321

    d. F7K75P

7. A company, Computer Bytes, sells computer equipment. You have been asked to design software that will input data from the keyboard, for each stock item and write the record to a sequential file. The product number for the stock item is 5 digits and has a modulus 10 check digit. The software must validate the input data so that invalid data is not accepted. A printed list is required of all the stock items in the file with the total number of records at the end of the report. The company stock over 500 items so the printed output must allow for multiple pages.

    Produce the following documentation:

    - Structure chart
    - File layout
    - List of the validation checks on input
    - List of the error messages
    - Screen layout
    - Print layout.

[This page is intentionally blank]