Table of Contents:

Moving Exercise

Introduction

An input file is provided with 2 records per customer. Information will be taken from each record to produce one output file with the selected data.

Data - Input File

This is for **Record A**

Field	**Type**	**Size**
Account Number	Alphanumeric	8
Account Name.		
Title	Alphanumeric	3
Initials.		
First initial	Alphanumeric	1
Middle initial	Alphanumeric	1
Surname	Alphanumeric	30
Date of Issue	Alphanumeric	8
Date of receipt	Alphanumeric	8
Balance	Numeric	10
Filler	Alphanumeric	10
Type	Alphanumeric	1

This is for **Record B**

Field	**Type**	**Size**
Account Number	Alphanumeric	8
Address.		
Number	Numeric	4
Street	Alphanumeric	20
Town	Alphanumeric	20
County	Alphanumeric	10
Postcode	Alphanumeric	10
Filler	Alphanumeric	7
Type	Alphanumeric	1

Data - Output File

Field	**Type**	**Size**
Account number	Alphanumeric	8
County	Alphanumeric	10
Balance	Numeric	9
Unused	Alphanumeric	52
Type	Alphanumeric	1

Processing Required

Records will be in order in the input file, record A followed by B followed by A etc. Output file should consist of the account number and balance taken from record A and the county taken from record B as well as an unused field of spaces and type containing a 0.

- The input file is [USERID].CBLPROG1.INPUT1
- You can write your own JCL or use [USERID].JCLLIB.TEST(PROG1JCL)
- Write this is DTB – it'll be fun.

The COBOL version of the program is [USERID].SOURCE.TEST(CBLPROG1), if it were me I'd start with re-writing the main proc logic, the moves.

Arithmetic Exercise

Introduction

In a department store that may or may not be running a sale on a particular day, a record is created for every item that is sold in any of its five departments. The records are input over a monthly period, in no particular sequence.

Data - Input File

Department	Numeric	1
Quantity	Numeric	4
Retail Price	Numeric	5
Sale Price	Numeric	5
Invoice Number	Alphanumeric	5
Sale indicator	Alphanumeric	60

Data - Good Output File

Department 1	Numeric	1
Total 1	Comp-3	7
Department 2	Numeric	1
Total 2	Comp-3	7
Department 3	Numeric	1
Total 3	Comp-3	7
Department 4	Numeric	1
Total 4	Comp-3	7
Department 5	Numeric	1
Total 5	Comp-3	7
Filler	Alphanumeric	55

Data - Bad Output File

Department	Numeric	1
Error	Alphanumeric	5 (value always Error)
Invoice Number	Alphanumeric	4
Filler	Alphanumeric	70

Processing Required

Each input record is to be validated as required in the section below.

Any invalid records will generate an error record on output file B, you can also add displays to identify which validation lead to the error being written. Valid records are to be accumulated into 1 of 5 department totals, the value being determined by multiplying the quantity by the appropriate price.

At the end of the input file, one record is to be produced on output file A with the department number and the total sales for each department.

Validation Required

1. The department number is in the range of 1-5
2. The indicator field can only contain the letter S or a space
3. Quantity, retail price and sale price are all numeric
4. Quantity and retail price are greater than 0
5. The first character of the invoice is A through R inclusive and the rest of the field is numeric
6. If the indicator field is blank then the sale price is 0
7. If the indicator field contains the letter ‘S’ then the sale price is present with the sale price less than the retail price.

Input JCL

[USERID].CBLPROG2.GOOD1 is a file that contains perfectly correct data – write yourself a test plan of what you think you need to check (the 7 points above are a clue) and then you can edit the file to create these scenarios e.g. changing one of the department numbers to 6 should fail – that would be a test.

[USERID].CBLPROG2.BAD1 is a file that already has errors in and will test your validation, if you're having problems testing it yourself you can use this file to hopefully narrow down where the issue is.

You can write your own JCL to execute the program but if you get stuck then you can use [USERID].JCLLIB.TEST(PROG2JCL).

I've created a linkparm member for you in [USERID].LINKPARM – it's based on you naming your program CBLPROG2 so if you don't then you'll have to change the linkparm accordingly. The linkparm is used for when you TSO BIND.

Subscripting/Indexing Exercise

Introduction

The way people have voted is recorded on records – the file should consist of one constituency (indicated by a four-character identifier). Each column holds a person's vote as a single digit in the range of 0-6 to indicate the party voted for.

The parties are as follows:
0 – Raving Loony Party
1 – Conservative
2 – Plaid Cymru
3 – Free Cleethorpes
4 – Labour
5 – Liberal
6 – Scottish National
7 – Spoilt votes

Data - Input File

Cols 1-4 is for the Constituency Identifier

Cols 5-76 is for the Votes
Each column could contain a vote in the range of 0-6. Consecutive columns are used, but a record could finish before column 76 with a space character which is not a spoilt vote)

Cols 77-80 is for Spaces

Processing

Count all the votes for each party.

The constituency ID will be input as a parameter in JCL. Any record that does not match the identifier will have its total number of votes added to Spoilt votes.

Any vote not in the range of 0-6 should also be in Spoilt votes. Produce, on a print file, a single page listing at the end of the run showing each party, Spoilt votes, the number of votes achieved and the winning party with *** next to the name.

Extraction Exercise

Introduction

In this exercise, you'll need to create an extract program based on the diagram below. This exercise precedes the next one.

Data - Good Input File

The records are input into the program in no particular sequence. It is **NOT** possible to have 2 or more input records with the same account number.

Account Number	Numeric	5
Initial	Alphanumeric	1
Surname	Alphanumeric	20
Credit/Debit Marker	Alphanumeric	1
Transaction Amount	Numeric	5
Transaction Date	Numeric	8
Transaction Details	Alphanumeric	20
Not Used	Alphanumeric	20

Data - Extracted Good Output File

The records are output from the program in no particular sequence. It is **NOT** possible to have 2 or more extracted good records with the same number.

Account Number	Alphanumeric	5
Credit/Debit Marker	Alphanumeric	1
Transaction Amount	Packed Decimal	5
Transaction Date	Numeric	8
Initial	Alphanumeric	1
Surname	Alphanumeric	20
Not Used	Alphanumeric	20

Processing

Each input record on the Good File will produce an output record on the Extracted Good File.

When the Credit/Debit Marker has the value ‘D’ then the Transaction Amount must be held as a negative value.

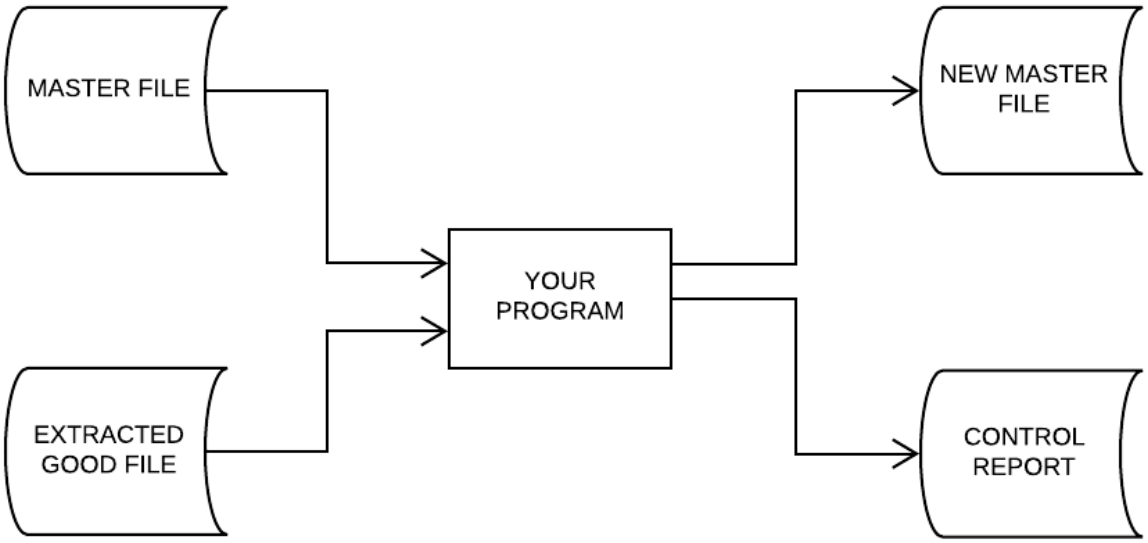
The Transaction Date must be held in the format - YYYYMMDD.

Blanks must be held in the Not Used field.

Update Exercise

Introduction

In this exercise, you'll need to create an update program based on the diagram below.



Data - Extracted Good Input File

The records are input into the program in **ascending** account number sequence. It is **NOT** possible to have 2 or more input records with the same account number.

Account Number	Alphanumeric	5
Credit/Debit Marker	Alphanumeric	1
Transaction Amount	Packed Decimal	5
Transaction Date	Numeric	8
Initial	Alphanumeric	1
Surname	Alphanumeric	20
Not Used	Alphanumeric	20

Data - Master Input File

The records are input into the program in **ascending** account number sequence. It is **NOT** possible to have 2 or more input records with the same account number.

Account Number	Alphanumeric	5
Initial	Alphanumeric	1
Surname	Alphanumeric	20
Balance	Packed Decimal	5
Historical Transactions	Occurs 5 Times	
Credit/Debit Marker	Alphanumeric	1
Transaction Amount	Numeric	5
Not Used	Alphanumeric	19

Data - New Master Output File

The records are output from the program in **ascending** account number sequence. It is **NOT** possible to have 2 or more output records with the same account number.

Account Number	Alphanumeric	5
Initial	Alphanumeric	1
Surname	Alphanumeric	20
Balance	Packed Decimal	5
Historical Transactions	Occurs 5 Times	
Credit/Debit Marker	Alphanumeric	1
Transaction Amount	Numeric	5
Not Used	Alphanumeric	19

Data - Control Report File Format

Type of Record	Total	Percentage
Unchanged Records	999	999.99
New Records	999	999.99
Updated Records	999	999.99
All Record Types	999	999.99

Processing

The records on the Extracted Good File and the Master File are to be processed together as a two file match program.

For each account number record on the Master File that does not have a matching record on the Extracted Good File will produce an unchanged record to be written out to the New Master File.

For each account number record on the Extracted Good File that does not have a matching record on the Master File will produce a new record to be written out to the New Master File.

- The Balance will hold the Transaction Amount.
- The Credit/Debit Marker and Transaction Amount will be held in the first occurrence of the Historical Transactions.
- Other occurrences of Historical Transactions must be held as spaces and zeroes respectively.
- Blanks must be held in the Not Used field.

For each matching account number record will produce an updated record to be written out to the New Master File.

- The Transaction Amount must be used to update the Balance
- The Credit/Debit Marker and Transaction Amount will be held in the first occurrence of the Historical Transactions.
- Other occurrences of Historical Transactions will be adjusted accordingly.

Finally, a control report will be produced as a summary.

VSAM Mayhem Exercise

Introduction

This is a simple report printing program, however, you'll need to convert the previous program's output into a VSAM file first.

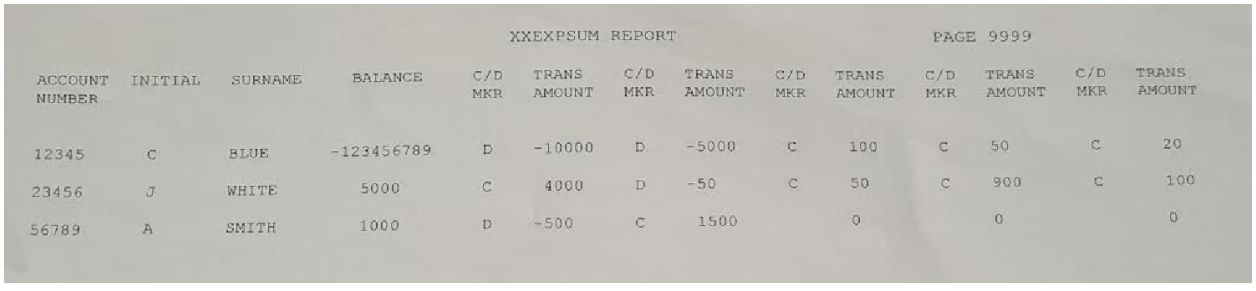
Data - New Master Input File

This is a **VSAM** file. The records are input into the program in **ascending** account number (primary key) sequence. It is **NOT** possible to have 2 or more output records with the same account number.

Account Number	Alphanumeric	5
Initial	Alphanumeric	1
Surname	Alphanumeric	20
Balance	Packed Decimal	5
Historical Transactions	Occurs 5 Times	
Credit/Debit Marker	Alphanumeric	1
Transaction Amount	Numeric	5
Not Used	Alphanumeric	19

Data - SYSOUT Output Report

Use the following format as depicted in the image below.



XXEXPSUM REPORT PAGE 9999

ACCOUNT NUMBER	INITIAL	SURNAME	BALANCE	C/D MKR	TRANS AMOUNT	C/D MKR	TRANS AMOUNT	C/D MKR	TRANS AMOUNT	C/D MKR	TRANS AMOUNT	C/D MKR	TRANS AMOUNT
12345	C	BLUE	-123456789	D	-10000	D	-5000	C	100	C	50	C	20
23456	J	WHITE	5000	C	4000	D	-50	C	50	C	900	C	100
56789	A	SMITH	1000	D	-500	C	1500		0		0		0

Processing

Each page of output has 3 heading lines (lines 1, 3 and 4). The page number will need to be updated for each new page. Each input record produces a detail line. A blank line must appear between each detail line. There will be a maximum of 54 lines (27 detail lines) per page.