```
i=1          # Counting upto 10.
while [ $i -le 10 ]
do
    echo $i
    i='expr $i + 1'
done
```

'%' for remainder

Escape Sequence	Meaning
\033[0m	Normal characters
\033[1m	Bold characters
\033[4m	Underlined characters
\033[5m	Blinking characters
\033[7m.	Reverse video characters.
\07	Bell
\n	New Line
\r	Carriage Return
\t	Tab key
\b	Back space (1 column)
\c	Same line

```
# RPwd1
trap "" 0 1 2 3
banner terminal
banner locked
read key
while true
do
  echo "Enter Password:\c"
  stty -echo
  read pw
  stty sane
  if [ "$pw" = icit]
  then
    break
  else
    echo wrong Password
  fi
done
```

```
# RPwd
old='stty -g'
stty -echo incle
echo "Enter Password:\c"
read pw
stty $old
if [ "$pw" = icit] then
  echo Correct Password.
else
  echo wrong Password
fi
```

unset Command is used to remove the function.
ie. unset functionname ↵

16 Shell Programming Project

Payroll Processing System
- Data Organisation
- Menus
- Report Formats
- Calculations

Working of The System

Program

Where Do You Go from Here...

Improve This Program...

* touch Command is used to create an empty file quickly. It doesn't allow you to store anything in a file. ie. touch filename ↵

UNIX - PROJECT

Shell Programming

So far we have written small but useful shell scripts. The idea was always to introduce some new concept or a subtlety of shell programming. Now that we are through with most of the features of shell programming let us put together an entire system using shell scripts. Development of this system would encompass all that we have learnt so far. This would help you in two ways:

(a) It would serve as a revision of all the concepts.
(b) It would help you visualise in which situation which concept has to be used while writing professional level scripts.

However, instead of handing you the entire software on a platter, I thought it worthwhile to give you the overall picture and getting you started and then leaving it for you to develop some components of the software. We would try to develop a payroll processing system. So here we go...

Payroll Processing System

Shivley & Brett is a pharmaceutical company engaged in manufacture of life saving drugs. The company was established in the year 1982 with an employee strength of 75, with 8 officers. Since its inception, it has grown steadily over the years and today boasts a turnover of Rs. 130 millions. Today it has a strength of 600 employees out of which 125 are of officer cadre. There are five departments in the company, viz., Manufacturing, Assembly, Stores, Accounts and Maintenance. It has been the policy of the company to develop a technologically advanced work environment. In tune with this policy,

the accounts department of the company has decided to computerise its payroll preparation. Since the payroll processing package for the company must be made to suit their requirements, the company has approached you to prepare the package. To begin with, the company has decided to implement computerised payroll processing only for the workers, with plans to include the officer staff only after the satisfactory completion of workers payroll processing.

Data Organisation

The workers in the company are divided into 5 categories, namely, Super skilled (SSK), Highly Skilled (HSK), Skilled (SKI), Semi skilled (SMS) and Unskilled (USK). The data about each worker in the company can be categorised as shown below:

General Data	Allowances	Deductions	Leave Record
Employee code	DA	Provident fund	Maximum CL
Name	HRA	ESI	Maximum ML
Department	CA	GIP	Maximum PL
Grade	CCA	Income tax	Cumulative CL
Sex	Special Pay 1	Profession tax	Cumulative ML
Address	Special Pay 2	Rent deduction	Cumulative PL
GPF no.	Gross Pay	LT loan installment	Cumulative LWP
GI scheme no.		ST loan installment	Attended days
ESI scheme no.		Special deduction 1	Monthly CL
Basic salary		Special deduction 2	Monthly ML
		Total deduction	Monthly PL
			Monthly LWP

This data can be organised into 2 different files:

(a) Employee master data file - This contains that information about the employees which is relatively permanent. (EMASTER.DBF)

(b) Employee transaction data file - This contains data which varies from month to month. (ETRAN.DBF)

For maintaining one to one correspondence between records in master data file and transaction data file, the Employee code and Department in which the employee works would be present in both the data files. The fields in each data file would be as under:

EMASTER.DBF	
Description	Field Name
Employee code	e_empcode
Name	e_empname
Sex	e_sex
Address	e_address
Name of city	e_city
Pin code	e_pin
Department	e_dept
Grade	e_grade
GPF. no.	e_gpf_no.
GI scheme no.	e_gis_no.
ESI scheme no.	e_esis_no.
CL allowed	e_max_cl
PL allowed	e_max_pl
ML allowed	e_max_ml
Basic salary	e_bs
Cumulative cl	e_cum_cl
Cumulative pl	e_cum_pl
Cumulative ml	e_cum_ml
Cumulative lwp	e_cum_lwp
Cumu. att.days	e_cum_att

ETRAN.DBF	
Description	Field Name
Employee code	t_empcode
Department	t_dept
Casual leave	t_cl
Medical leave	t_ml
Provi. leave	t_ pl
LWP	t_lwp
DA	t_da
HRA	t_hra
CA	t_ca
CCA	t_cca
Special pay 1	t_sppay_1
Special pay 2	t_sppay_2
Gross salary	t_gs
GPF	t_gpf
GIS	t_gis
ESIS	t_esis
Income tax	t_inc_tax
Profession tax	t_prof_tax
Rent deduction	t_rent_ded
Long term loan	t_lt_loan
Short term loan	t_st_loan
Special ded. 1	t_spded_1
Special ded. 2	t_spded_2
Total deduction	t_tot_ded
Net pay	t_net_pay

Menus

The different menus to be developed in this system are as follows:

PAYROLL SYSTEM
MAIN MENU

Database operations
Reports
Exit
Your Choice?

PAYROLL SYSTEM
DATA BASE OPERATIONS

Master File Data Entry
Transaction Data Entry
Return to main menu
Your Choice?

PAYROLL SYSTEM
MASTER DATA ENTRY

Add record
Modify record
Delete record
Retrieve record
Return to main menu
Your choice?

PAYROLL SYSTEM
TRAN. DATA ENTRY

Add record
Modify record
Delete record
Retrieve record
Return to main menu
Your choice?

PAYROLL SYSTEM
REPORTS MENU

Mailing Labels
Leave Status Report
Paysheet Printing
Summary Payroll Sheet
Return to main menu
Your choice?

PAYROLL SYSTEM
SYSTEM MAINT. MENU

Close month
Close year & reorganise
Return to main menu

Your choice?

Report Formats

The formats of the different reports to be printed are as follows:

Mailing List

This report displays the entire mailing list of employees.

Clarence Elmsworth
Blandings Castle
Kalyan
411002

Jason Bourne
Whispering Palms, Madh Island
Bombay
400064

John Galt
A-2, Manish Nagar, Andheri
Bombay
400054

Michael Havilland
22, Shangri-la, Bandra
Banglore
560050

Leave Status Report

This report shows the status of leaves taken by an employee during the current year in terms of allowed leaves, leaves availed so far (cumulative) and leaves in balance.

Name : John Galt		Empcode:A10	Grade:HSK	Month:Mar96
CL allowed	ML allowed	PL allowed		
12	15	5		
Cum. CL	Cum. ML	Cum.PL	CumLWP	Cum. Att. days
3	4	0	0	82
Balance CL	Balance ML	Balance PL		
9	11	5		

Name : Jason Bourne		Empcode:A01	Grade: SSK	Month:Mar96
CL allowed	ML allowed	PL allowed		
15	20	5		
Cum. CL	Cum. ML	Cum. PL	Cum. LWP	Cum Att.days
1	0	0	0	102
Balance CL	Balance ML	Balance PL		
14	20	5		

Payslip

This report generates payslips of employees working in the factory. Note that such reports are usually printed on pre-printed stationary. In this system it has been printed only on the screen.

SHIVLEY & BRETT PVT. LTD.										
Emp.code:A10		Sex: Male			Grade: HSK			Month:Mar		
Name: John Galt					Department: Assembly					
GPF No. 6132/A		GIS No.P6329			ESIS No. P6452					
Normal Days		Casual Leave		Medical Leave	Prov. Leave		LWP	Attended days		
31		2		1	0		0	28		
BS	DA		HRA	CA	CCA	SP 1	SP 2	GS		
520.0	1040.00		130.0	52.00	52.00	45.00	0.00	1839.00		
GPF	ESIS	GIS	IT	PT	Rent	Loan1	Loan2	S.D.1	SD.2	Tot.
156.0	75.00	115.0	0.00	20.00	50.00	0.00	75.00	0.00	0.00	491.0
Net Pay										
Rs. 1348.00								Receiver's signature		

Summary Payroll Sheet

This report gives department wise payment made for a particular month for all employees in each department.

Summary Payroll Sheet

March 1996

Department	Total Employees	Gross Earnings	Gross Deductions	Net Payment
MFG	132	32,040	3,530	28,510
ASSLY	67	21,030	2,850	18,180
STORES	19	17,550	1,870	15,680
ACCTS	22	18,490	2,115	16,375
MAINT	30	24,555	3,335	21,220

Calculations

The following table shows percentages used for calculation of various allowances and deductions for different grades of employees.

Grade	DA	HRA	CA	CCA	GPF	ESIS	GIS	PT
	% of BS	% of BS	% of BS	% of BS	**			
SSK	200 %	30 %	10 %	10 %	10 %	Rs. 100	Rs. 115	Rs. 50
HSK	200 %	25 %	10 %	10 %	10 %	Rs. 100	Rs. 115	Rs. 50
SKI	100 %	25 %	10 %	10 %	10 %	Rs. 100	Rs. 115	Rs. 50
SMS	100 %	20 %	10 %	10 %	10 %	Rs. 100	Rs. 115	Rs. 20
USK	175 %	18 %	Rs. 150	10 %	10 %	Rs. 100	Rs. 115	-----

**** % of BS + DA

Quite naturally, this big a system cannot be implemented using a single shell script. In fact there would be several of them each doing a specific job and interacting with one another to work like a system as a whole. This once again is in tune with the Unix philosophy - build small parts, make them do their job well and then combine them, link them to build a powerful, robust system.

Given below is a list of the various shell scripts that are developed in this system along with the purpose of each. This will help you to keep track of the system as you read the listings given in the subsequent pages.

Shell Script	Purpose
paymain.prg	Does some initial house-keeping, Displays Main Menu and branches control to appropriate sub-menu.
writecentre	Writes a given string in the center of a given row either in Bold, Normal or Reverse video.
writerc	Writes a given string at the given row, column either in Bold, Normal or Reverse video.
dboper.prg	Displays Database Operations Menu and branches control to either Master or Transaction Data Entry Menu.
reports.prg	Displays Reports Menu and branches control to generate the appropriate report.
sysmnt.prg	Displays System Maintenance Menu and branches control to carry out appropriate house-keeping job.
mde.prg	Displays Master Data Entry Menu and branches control to carry out appropriate operation on employee master.
tde.prg	Displays Transaction Data Entry Menu and branches control to carry out appropriate operation on employee transaction file.
madd.prg	Adds new records to master file.
mmodi.prg	Modifies an existing record in master file.

Shell Script	Purpose
mdel.prg	Deletes an existing record from master file.
mret.prg	Retrieves record from master file and displays it on the screen.
tadd.prg	Adds new records to employee transaction file.
maillbl.prg	Prints mailing labels.
payprint.prg	Prints monthly payslips for employees on screen.
spaysheet.prg	Prints department-wise summary payroll sheet.
lsr.prg	Generates leave status report.
clmonth.prg	Closes the monthly transaction file.
clyear.prg	Closes the yearly transactions, updates master and reorganizes files and variables.

The overall breakup of the system on a file by file basis and their hierarchy of calling is given in Figure 16.1. Files which are general and are called by several other scripts are shown separately.

With this much detailing I suppose you would be able to follow the shell script listings given below. They have been suitably commented to help you understand the underlying logic.

Working of The System

The user just has to add records to the master and the monthly transaction file. While adding records to master file the program avoids any duplication of records since there must be a unique record for each employee. Also while adding records to the monthly transaction file the program makes sure that a record doesn't get added to the transaction file unless there is a corresponding record in the master file. Here too, the program prevents any unintentional or otherwise duplication of records.

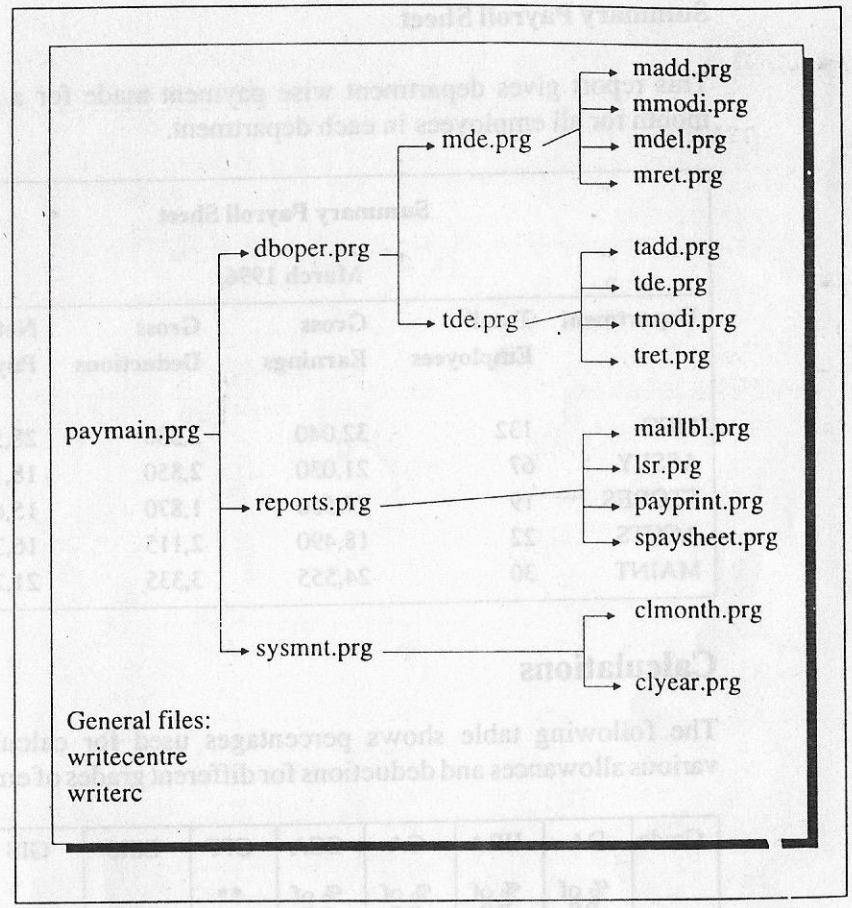

Figure 16.1

At the end of each month the user should generate all the reports through the reports menu. Before starting data entry for new transactions for a new month the user must specifically close that month's transactions through the 'Close month' option from the System Maintenance menu. At the end of the financial year the 'Close year' option from the System Maintenance menu should be exercised. It generates a transaction file which is a combination of all monthly transaction files in that financial year. It has been assumed that the financial year runs from April through March of next year.

Program

paymain.prg

```
# paymain.prg
# Does some initial house-keeping, Displays Main Menu
# and branches control to appropriate sub-menu.

MASTER=$HOME/emaster.dbf
TRAN=$HOME/etran.dbf
export MASTER TRAN

# check if master and transaction files exist
if [ ! -f $MASTER ]
then
    touch $MASTER
fi
if [ ! -f $TRAN ]
then
    touch $TRAN
fi

while true
do
    # display Main Menu
    clear
    writecentre "Payroll Processing System" 7 "B"
    writecentre "Main Menu" 8 "N"
    writerc "\033[1mD\033[0matabase operations" 10 30 "N"
    writerc "\033[1mR\033[0meports" 11 30 "N"
    writerc "\033[1mS\033[0mystem maintenance" 12 30 "N"
    writerc "\033[1mE\033[0mxit" 13 30 "N"
    writerc "Your choice? " 15 30 "N"

    # receive user's choice
    choice=""
```



```
    stty -icanon min 0, time 0
    while [ -z "$choice" ]
    do
        read choice
    done
    stty sane

    # branch off to appropriate menu
    case "$choice" in
        [Dd]) dboper.prg ;;
        [Rr]) reports.prg ;;
        [Ss]) sysmnt.prg ;;
        [Ee]) clear
              exit ;;
        *) echo \007 ;;
    esac
done
```

writecentre

```
# writecentre
# Writes a given string in the center of a given row
# either in Bold, Normal or Reverse video.

# check whether called properly
if [ $# -ne 3 ]
then
    echo improper arguments
    exit
fi

# position cursor
str=$1
row=$2
attr=$3
```

```
length='echo $str | wc -c'
col='expr \( 80 - $length \) / 2'
tput cup $row $col

# display string in Bold, Normal or Reverse video
case $attr in
    [bB])echo -n "\033[1m$str" ;;
    [nN])echo -n $str ;;
    [rR])echo -n "\033[7m$str" ;;
esac
echo -n "\033[0m"
```

writerc

```
# writerc
# Writes a given string at the given row, column
# either in Bold, Normal or Reverse video.

# check whether called properly
if [ $# -ne 4 ]
then
    echo improper arguments
    exit
fi

# position cursor
str="$1"
row=$2
col=$3
attr=$4
tput cup $row $col

# display string in Bold, Normal or Reverse video
case $attr in
    [bB])echo -n "\033[1m$str" ;;
    [nN])echo -n "$str" ;;
```



```
    [rR])echo -n "\033[7m$str" ;;
esac
echo -n "\033[0m"
```

dboper.prg

```
# dboper.prg
# Displays Database Operations Menu and branches control to
# either Master or Transaction Data Entry Menu.

while true
do
    # display Database Operations Menu
    clear
    writecentre "Payroll Processing System" 7 "B"
    writecentre "Data Base Operation" 8 "B"
    writerc "\033[1mM\033[0master File Data Entry"  10 30 "N"
    writerc "\033[1mT\033[0mransaction Data Entry" 11 30 "N"
    writerc "\033[1mR\033[0meturn to Main Menu" 12 30 "N"
    writerc "Your choice? " 15 30 "N"

    # receive user's choice
    choice=""
    stty -icanon min 0, time 0
    while [ -z "$choice" ]
    do
        read choice
    done
    stty sane

    # check user's choice and branch off to
    # appropriate Data Entry Menu
    case "$choice" in
        [Mm]) mde.prg ;;
        [Tt]) tde.prg ;;
        [Rr]) clear
```

```
                break ;;
            *) echo  \007 ;;
        esac
done
```

reports.prg

```
# reports.prg
# Displays Reports Menu and branches control to generate the
# appropriate report.

while true
do
    # display Reports Menu
    clear
    writerc "\033[36mPayroll Processing System\033[37m" 7 27 "B"
    writerc "\033[36mReports Menu\033[37m" 8 33 "N"
    writerc "\033[1mM\033[0mailing Labels" 10 30 "N"
    writerc "\033[1mL\033[0meave Status Report" 11 30 "N"
    writerc "\033[1mP\033[0maysheet Printing" 12 30 "N"
    writerc "\033[1mS\033[0mummary Payroll Sheet" 13 30 "N"
    writerc "\033[1mR\033[0meturn to main menu" 14 30 "N"
    writerc "Your choice? " 16 30 "N"

    # receive user's choice
    choice=""
    stty -icanon min 0, time 0
    while [ -z "$choice" ]
    do
        read choice
    done
    stty sane

    # check user's choice and branch off to
    # generate an appropriate report
    case "$choice" in
```



```
        [Mm]) maillbl.prg ;;
        [Ll]) lsr.prg ;;
        [Pp]) payprint.prg ;;
        [Ss]) spaysheet.prg ;;
        [Rr]) clear
              break ;;
        *) echo  \007 ;;
    esac
done
```

sysmnt.prg

```
# sysmnt.prg
# Displays System Maintenance Menu and branches control to
# carry out appropriate house-keeping job.

while true
do
    # display System Maintenance Menu
    clear
    writerc "\033[36mPayroll Processing System\033[37m" 7 27 "B"
    writerc "\033[32mSystem Maintenance Menu\033[37m" 8 28 "B"
    writerc "c\033[1mL\033[0mose month" 10 30 "N"
    writerc "\033[1mC\033[0mlose year & reorganise" 11 30 "N"
    writerc "\033[1mR\033[0meturn to main menu" 12 30 "N"
    writerc "Your choice? " 14 30 "N"

    # receive user's choice
    choice=""
    stty -icanon min 0, time 0
    while [ -z "$choice" ]
    do
        read choice
    done
    stty sane
```

```
    # check user's choice and branch off to
    # carry out appropriate house-keeping job
    case "$choice" in
        [Ll]) clmonth.prg ;;
        [Cc]) clyear.prg ;;
        [Rr]) clear
              break ;;
        *) echo  \007 ;;
    esac
done
```

mde.prg

```
# mde.prg
# Displays Master Data Entry Menu and branches control to
# carry out appropriate operation on employee master.

while true
do
    # display Master Data Entry Menu
    clear
    writecentre "Payroll Processing System" 7 "B"
    writecentre "Master File Data Entry" 8 "B"
    writerc "\033[1mA\033[0mdd records " 10 30 "N"
    writerc "\033[1mM\033[0modify records" 11 30 "N"
    writerc "\033[1mD\033[0melete record" 12 30 "N"
    writerc "\033[1mR\033[0metrieve record" 13 30 "N"
    writerc "r\033[1mE\033[0mturn" 14 30 "N"
    writerc "Your choice? " 16 30 "N"

    # receive user's choice
    choice=""
    stty -icanon min 0, time 0
    while [ -z "$choice" ]
    do
```



```
        read choice
    done
    stty sane

    # check user's choice and branch off to perform
    # appropriate operation on employee master
    case "$choice" in
        [Aa]) madd.prg ;;
        [Mm]) mmodi.prg ;;
        [Dd]) mdel.prg ;;
        [Rr]) mret.prg ;;
        [Ee]) exit ;;
        *) echo  \007 ;;
    esac
done
```

tde.prg

```
# tde.prg
# Displays Transaction Data Entry Menu and branches control to
# carry out appropriate operation on employee transaction file.

while true
do
    # display Transaction Data Entry Menu
    clear
    writecentre "Payroll Processing System" 7 "B"
    writecentre "Transaction Data Entry" 8 "B"
    writerc "\033[1mA\033[0mdd records " 10 30 "N"
    writerc "\033[1mM\033[0modify records" 11 30 "N"
    writerc "\033[1mD\033[0melete record" 12 30 "N"
    writerc "\033[1mR\033[0metrieve record" 13 30 "N"
    writerc "r\033[1mE\033[0mturn" 14 30 "N"
    writerc "Your choice? " 16 30 "N"

    # receive user's choice
```

```
    choice=""
    stty -icanon min 0, time 0
    while [ -z "$choice" ]
    do
        read choice
    done
    stty sane

    # check user's choice and branch off to perform
    # appropriate operation on employee transaction file
    case "$choice" in
        [Aa]) tadd.prg ;;
        [Mm]) tmodi.prg ;;
        [Dd]) tdel.prg ;;
        [Rr]) tret.prg ;;
        [Ee]) clear
              exit ;;
        *) echo \007 ;;
    esac
done
```

madd.prg

```
# madd.prg
# Adds new records to master file.

another=y
t='tty'

while [ "$another" = y -o "$another" = Y ]
do
    clear
    writecentre "Payroll Processing System" 1 "B"
    writecentre "Add Records - Master File" 2 "B"

    writerc "Employee Code: \c" 4 10 "B"
```



```
read e_empcode
if [ -z "$e_empcode" ]
then
    exit
fi

# check if such an employee code already exists
grep \^$e_empcode: $MASTER > /dev/null
if [ $? -eq 0 ]
then
    writerc "Code already exists. Press any key..." 20 10 "N"
    read key
    continue
fi

# read values of various fields
writerc "Name of Employee: \c" 5 10 "B"
read e_empname
writerc "Sex: \c" 6 10 "B"
read e_sex
writerc  "Address: \c" 7 10 "B"
read e_address
writerc "Name of city: \c" 8 10 "B"
read e_city
writerc "Pin code number: \c" 9 10 "B"
read e_pin
writerc "Department: \c" 10 10 "B"
read e_dept
writerc "Grade: \c" 11 10 "B"
read e_grade
writerc "GPF no. \c" 12 10 "B"
read e_gpf_no
writerc "GI scheme no.: \c" 13 10 "B"
read e_gis_no
writerc "ESI scheme no.: \c" 14 10 "B"
read e_esis_no
writerc "CL allowed: \c" 15 10 "B"
```

```
        read e_max_cl
        writerc "PL allowed: \c" 16 10 "B"
        read e_max_pl
        writerc "ML allowed: \c" 17 10 "B"
        read e_max_ml
        writerc "Basic Salary: \c" 18 10 "B"
        read e_bs

        e_cum_cl=0
        e_cum_pl=0
        e_cum_ml=0
        e_cum_lwp=0
        e_cum_att=0

        # write new record into employee master file
        echo $e_empcode:$e_empname:$e_sex:$e_address:$e_city:
        $e_pin:$e_dept:$e_grade:$e_gpf_no:$e_gis_no:$e_esis_no:
        $e_max_cl:$e_max_pl:$e_max_ml:$e_bs:$e_cum_cl:$e_cum_pl:
        $e_cum_ml:$e_cum_lwp:$e_cum_att | dd conv=ucase
            2> /dev/null >> $MASTER

        writerc "Add another y/n \c" 20 10 "N"
        read another
done
```

mmodi.prg

```
# mmodi.prg
# Modifies an existing record in master file.

another=y

while [ "$another" = y -o "$another" = Y ]
do
    clear
    writecentre "Payroll Processing System" 1  "B"
```



```
    writecentre "Modify Records - Master File" 2 "B"

    writerc "Employee Code: " 4 10 "B"
    read e_empcode
    if [ -z "$e_empcode" ]
    then
        exit
    fi

    grep \^$e_empcode: $MASTER > /dev/null
    if [ $? -ne 0 ]
    then
        writerc "Employee code does not exist. Press any key..." 10 10 "B"
        read key
        continue
    fi

    # transfer all other records to a temporary file
    grep -v \^$e_empcode: $MASTER > /tmp/emaster.mmm

    mline=`grep \^$e_empcode: $MASTER`
    oldIFS="$IFS"
    IFS=':'
    set -- $mline

    writerc "Name: $2" 5 10 "N"
    read e_empname

    # if no change is made in employee name
    if [ -z "$e_empname" ]
    then
        e_empname=$2
    fi

    writerc "sex: $3" 6 10 "N"
    read e_sex
```

```
# if no change is made in employee sex
if [ -z "$e_sex" ]
then
    e_sex=$3
fi

writerc "Address: $4" 7 10 "N"
read e_address
if [ -z "$e_address" ]
then
    e_address=$4
fi

writerc "City: $5" 8 10 "N"
read e_city
if [ -z "$e_city" ]
then
    e_city=$5
fi

writerc "Pin code No: $6" 9 10 "N"
read e_pin
if [ -z "$e_pin" ]
then
    e_pin=$6
fi

writerc "Department: $7" 10 10 "N"
read e_dept
if [ -z "$e_dept" ]
then
    e_dept=$7
fi

writerc "Grade: $8" 11 10 "N"
read e_grade
if [ -z "$e_grade" ]
```



```
then
    e_grade=$8
fi

writerc "GPF no: $9" 12 10 "N"
read e_gpf_no
if [ -z "$e_gpf_no" ]
then
    e_gpf_no=$9
fi

shift 9

writerc  "GI scheme no: $1" 13 10 "N"
read e_gis_no
if [ -z "$e_gis_no" ]
then
    e_gis_no=$1
fi

writerc "ESI sscheme no: $2" 14 10 "N"
read e_esis_no
if [ -z "$e_esis_no" ]
then
    e_esis_no=$2
fi

writerc "CL allowed: $3" 15 10 "N"
read e_max_cl
if [ -z "$e_max_cl" ]
then
    e_max_cl=$3
fi

writerc "PL allowed: $4" 16 10 "N"
read e_max_pl
if  [ -z "$e_max_pl" ]
```

```
then
    e_max_pl=$4
fi

writerc "ML allowed: $5" 17 10 "N"
read e_max_ml
if [ -z "$e_max_ml" ]
then
    e_max_ml=$5
fi

writerc "Basic salary: $6" 18 10 "N"
read e_bs
if [ -z "$e_bs" ]
then
    e_bs=$6
fi

e_cum_cl=$7
e_cum_pl=$8
e_cum_ml=$9

shift 9
e_cum_lwp=$1
e_cum_att=$?

IFS="$oldIFS"

# append modified record to employee master file
echo $e_empcode:$e_empname:$e_sex:$e_address:$e_city:
$e_pin:$e_dept:$e_grade:$e_gpf_no:$e_gis_no:$e_esis_no:
$e_max_cl:$e_max_pl:$e_max_ml:$e_bs:$e_cum_cl:$e_cum_pl:
$e_cum_ml:$e_cum_lwp:$e_cum_att | dd conv=ucase
    2> /dev/null >> /tmp/emaster.mmm

# move new master over top of original
mv /tmp/emaster.mmm $MASTER
```



```
    writerc "Modify Another y/n " 20 20 "N"
    read another
done
```

mdel.prg

```
# mdel.prg
# Deletes an existing record from master file.

another=y
while [ "$another" = y ]
do
    clear
    writecentre "Payroll Processing System" 1 "B"
    writecentre "Delete Records - Master File" 3 "B"

    writerc "Employee Code to Delete: \c" 6 10 "B"
    read e_empcode
    if [ -z "$e_empcode" ]
    then
        exit
    fi

    # check whether employee code exists
    grep -y \^$e_empcode: $MASTER > /dev/null
    if [ $? -ne 0 ]
    then
        writerc "Employee code does not exist... Press any key" 10 10 "B"
        read key
        continue
    fi

    # rewrite all other records into a new file
    grep -vy \^$e_empcode: $MASTER > /tmp/emaster.ddd
    mv /tmp/emaster.ddd $MASTER
```

```
    # check whether there is a corresponding employee in transaction file
    grep -y \^$e_empcode: $TRAN  >  /dev/null
    if [ $? -eq 0 ]
    then
        # eliminate the corresponding record from transaction file too
        grep  -vy  \^$e_empcode:  $TRAN  >  /tmp/etran.ddd
        mv  /tmp/emaster.ddd  $TRAN
    fi

    writerc "Delete another y/n " 16 15 "B"
    read another
done
```

mret.prg

```
# mret.prg
# Retrieves record from master file and display it on the screen.

another=y

while [ "$another" = y -o "$another" = Y ]
do
    clear
    writecentre "Payroll maintenance System" 1  "B"
    writecentre " Retrieve Records - Master File" 2 "B"

    writerc "Employee Code: " 4 10 "B"
    read e_empcode

    if [ -z "$e_empcode" ]
    then
        exit
    fi

    # search employee code in master file
```



```
    grep  \^$e_empcode:  $MASTER  >  /dev/null

    # if employee code doesn't exist
    if [ $? -ne 0 ]
    then
        writerc "Employee code does not exist. Press any key..." 10 10 "B"
        read key
        continue
    fi

    # separate out field values
    mline=`grep  \^$e_empcode:  $MASTER`
    oldIFS="$IFS"
    IFS=':'
    set -- $mline

    # display field values on screen
    writerc "Name: $2" 5 10 "N"
    writerc "sex: $3" 6 10 "N"
    writerc "Address: $4" 7 10 "N"
    writerc "City: $5" 8 10 "N"
    writerc "Pin code No: $6" 9 10 "N"
    writerc "Department: $7" 10 10 "N"
    writerc "Grade: $8" 11 10 "N"
    writerc "GPF no: $9" 12 10 "N"
    shift 9
    writerc "GI scheme no: $1" 13 10 "N"
    writerc "ESI sscheme no: $2" 14 10 "N"
    writerc "CL allowed: $3" 15 10 "N"
    writerc "PL allowed: $4" 16 10 "N"
    writerc "ML allowed: $5" 17 10 "N"
    writerc "Basic salary: $6" 18 10 "N"

    IFS="$oldIFS"
    writerc "Retrieve another y/n " 20 20 "N"
    read another
done
```

tadd.prg

```
# tadd.prg
# Adds new records to employee transaction file.

clear
another=y
t='tty'
IFScolon=":"
IFSspace="$IFS"

# percentages used for calculation, according to employee grade
SSK="200 30 10 10 10 75 115 20"
HSK="200 25 10 10 10 75 115 20"
SKI="100 25 10 10 10 75 115 15"
SMS="100 22 10 10 10 75 115 15"
USK="17 20 10 10 10 75 115 0"

while [ "$another" = y -o "$another" = Y ]
do
    clear
    writecentre "Payroll Processing System" 1 "B"
    writecentre "Add Records - Tran. File" 2 "B"

    writerc "Employee Code: " 4 10 "B"
    read t_empcode
    if [ -z "$t_empcode" ]
    then
        exit
    fi

    mline='grep \^$t_empcode: $MASTER'
    mfound=$?

    # if employee code is not found in master
    if [ $mfound -ne 0 ]
```



```
    then
        writecentre "Corresponding Master record absent" 7 "N"
        writecentre "Press any key..." 88888888 "N"
        read key
        continue
    fi

    grep \^$t_empcode: $TRAN > /dev/null
    tfound=$?

    # if employee code already exists in transaction file
    if [ $tfound -eq 0 ]
    then
        writecentre "Already exists, Cannot duplicate" 7 "N"
        writecentre "Press any key..." 8 "N"
        read key
        continue
    fi

    # read values of various fields
    writerc "Department: " 5 10 "B"
    read t_dept
    writerc "Casual leave: " 6 10 "B"
    read t_cl
    writerc "Medical leave: " 7 10 "B"
    read t_ml
    writerc "Provisional leave: " 8 10 "B"
    read t_pl
    writerc "LWP: " 9 10 "B"
    read t_lwp
    writerc "Special pay 1: " 10 10 "B"
    read t_sppay_1
    writerc "Special pay 2: " 11 10 "B"
    read t_sppay_2
    writerc "Income tax: " 12 10 "B"
    read t_inc_tax
    writerc "Rent deduction: " 13 10 "B"
```

```
read t_rent_ded
writerc "Long term loan: " 14 10 "B"
read t_lt_loan
writerc "Short term loan: " 15 10 "B"
read t_st_loan
writerc "Special ded. 1: " 16 10 "B"
read t_spded_1
writerc "Special ded. 2: " 17 10 "B"
read t_spded_2

# extract grade and basic salary from master
grade=`echo $mline | cut -d":" -f8`
bs=`echo $mline | cut -d":" -f15`

# calculate various allowances and gross salary
set -- `eval echo \\$$grade`
t_da=`echo "scale=2\n$1 / 100.0 * $bs" | bc`
t_hra=`echo "scale=2\n$2 / 100.0 * $bs" | bc`
t_ca=`echo "scale=2\n$3 / 100.0 * $bs" | bc`
t_cca=`echo "scale=2\n$4 / 100.0 * $bs" | bc`
t_gs=`echo "scale=2\n$bs + $t_da + $t_hra + $t_ca + $t_cca +
        $t_sppay_1 + $t_sppay_2" | bc`

# calculate various deductions
t_gpf=`echo "scale=2\n$5 / 100.0 * ( $bs + $t_da )" | bc`
t_esis=$6
t_gis=$7
t_prof_tax=$8
t_tot_ded=`echo "scale=2\n$t_gpf + $t_esis + $t_gis + $t_prof_tax +
            $t_inc_tax + $t_lt_loan + $t_st_loan + $t_rent_ded +
            $t_spded_1 + $t_spded_2" | bc`

# calculate net salary
t_net_pay=`echo "scale=2\n$t_gs - $t_tot_ded" | bc`

# write new record to transaction file
echo $t_empcode:$t_dept:$t_cl:$t_ml:$t_pl:$t_lwp:$t_da:$t_hra:
```



```
$t_ca:$t_cca:$t_sppay_1:$t_sppay_2:$t_gs:$t_gpf:$t_gis:$t_esis:
$t_inc_tax:$t_prof_tax:$t_rent_ded:$t_lt_loan:$t_st_loan:
$t_spded_1:$t_spded_2:$t_tot_ded:$t_net_pay | dd conv=ucase
    2> /dev/null >> $TRAN

# find number of days in current month
days="31 28 31 30 31 30 31 31 30 31 30 31"
months=`date '+%m'`
totdays=`echo $days | cut -d" " -f $months`

IFS="$IFScolon"
exec < $MASTER

# read each record from master file
while read e_empcode e_empname e_sex e_address e_city e_pin
    e_dept e_grade e_gpf_no e_gis_no e_esis_no e_max_cl
    e_max_pl e_max_ml e_bs e_cum_cl e_cum_pl e_cum_ml
    e_cum_lwp e_cum_att
do
    IFS="$IFSspace"

    if [ $e_empcode = $t_empcode ]
    then
        # update cumulative leaves fields
        e_cum_cl=`expr $e_cum_cl + $t_cl`
        e_cum_ml=`expr $e_cum_ml + $t_ml`
        e_cum_pl=`expr $e_cum_pl + $t_pl`
        e_cum_lwp=`expr $e_cum_lwp + $t_lwp`
        net_days=`expr $totdays - $t_cl - $t_ml - $t_pl - $t_lwp`
        e_cum_att=`expr $e_cum_att + $net_days`
    fi

    # write record to master file
    echo $e_empcode:$e_empname:$e_sex:$e_address:$e_city:
    $e_pin:$e_dept:$e_grade:$e_gpf_no:$e_gis_no:$e_esis_no:
    $e_max_cl:$e_max_pl:$e_max_ml:$e_bs:$e_cum_cl:$e_cum_pl:
    $e_cum_ml:$e_cum_lwp:$e_cum_att | dd conv=ucase
```

```
        2> /dev/null >> /tmp/master.aaa
        IFS="$IFScolon"
    done

    mv /tmp/master.aaa  $MASTER
    exec < $t
    IFS="$IFSspace"
    writerc "Add another y/n " 23 10 "N"
    read another
done
```

maillbl.prg

```
# maillbl.prg
# Prints mailing labels.

clear
t=`tty`

writerc "Screen/Printer " 10 20 "B"
read ans
writerc "Please wait..." 12 22 "B"

exec < $MASTER

while true
do
    read line1

    # if record read successfully
    if [ $? -eq 0 ]
    then
        # separate out relevant fields
        name1=`echo $line1 | cut -d":" -f 2`
        add1=`echo $line1 | cut -d":" -f 4`
```



```
        city1=`echo $line1 | cut -d":" -f 5`
        pin1=`echo $line1 | cut -d":" -f 6`

        # calculate lengths of various fields
        ln1=`echo $name1 | wc -c`
        la1=`echo $add1 | wc -c`
        lc1=`echo $city1 | wc -c`
        lp1=`echo $pin1 | wc -c`

        # calculate blanks to be padded
        bn1=`expr 40 - $ln1`
        ba1=`expr 40 - $la1`
        bc1=`expr 40 - $lc1`
        bp1=`expr 40 - $lp1`

        # pad blanks after name
        count=1
        while [ $count -le $bn1 ]
        do
            name1="$name1 "
            count=`expr $count + 1`
        done

        # pad blanks after address
        count=1
        while [ $count -le $ba1 ]
        do
            add1="$add1 "
            count=`expr $count + 1`
        done

        # pad blanks after city
        count=1
        while [ $count -le $bc1 ]
        do
            city1="$city1 "
            count=`expr $count + 1`
```

```
        done

        # pad blanks after pin
        count=1
        while [ $count -le $bp1 ]
        do
            pin1="$pin1 "
            count=`expr $count + 1`
        done
    else
        break
    fi

    # read another record from file
    line2=""
    read line2

    # separate out relevant fields
    name2=`echo $line2 | cut -d":" -f 2`
    add2=`echo $line2 | cut -d":" -f 4`
    city2=`echo $line2 | cut -d":" -f 5`
    pin2=`echo $line2 | cut -d":" -f 6`

    # write fields from 2 records side by side
    echo "$name1 $name2" >> mail.lbl
    echo "$add1 $add2" >> mail.lbl
    echo "$city1 $city2" >> mail.lbl
    echo "$pin1 $pin2" >> mail.lbl
    echo >> mail.lbl

done

exec < $t

if [ "$ans" = S -o "$ans" = s ]
then
    echo
```



```
    pg mail.lbl    # display mailing labels
else
    lpr mail.lbl  # print mailing labels
fi

rm mail.lbl
```

payprint.prg

```
# payprint.prg
# Prints monthly payslips for employees on SCREEN.
# In practice the payslips are printed on
# pre-printed computer stationery. Hence printing on
# printer would have to be planned according to the
# page layout of the pre-printed form.

clear

writerc "Screen/Printer " 10 20 "B"
read ans

# calculate number of days in current month
month=`date '+%B'`
days="31 29 31 30 31 30 31 31 30 31 30 31"
tmp=`date '+%m'`
mdays=`echo $days | cut -d" " -f $tmp`

another=y
t=`tty`
IFSspace="$IFS"

while [ "$another" = y ]
do
    clear

    writerc "Employee Code: " 4 10 "B"
```

```
read empcode

if [ -z "$empcode" ]
then
    exit
fi

# search the employee code
grep \^$empcode: $MASTER > /dev/null

# if the search fails
if [ $? -ne 0 ]
then
    writerc "Employee code does not exist. Press any key..." 10 10 "B"
    read key
    clear
    continue
fi

# build a horizontal dashed line
dln="-"
count=0
while [ $count -lt 78 ]
do
    dln="$dln-"
    count=`expr $count + 1`
done

clear
writerc "$dln" 0 1 "B"
writecentre "Shivley & Brett Pvt. Ltd." 1 "B"
writerc "$dln" 2 1 "B"

# set standard input to master file
exec < $MASTER

IFS=":"
```



```
# read till desired record is encountered
while read e_empcode e_empname e_sex e_address e_city e_pin
    e_dept e_grade e_gpf_no e_gis_no e_esis_no e_max_cl
    e_max_pl e_max_ml e_bs e_cum_cl e_cum_pl e_cum_ml
    e_cum_lwp e_cum_att
do
    if [ "$empcode" = "$e_empcode" ]
    then
        break
    fi
done

# set standard input to transaction file
exec < $TRAN

# read till desired record is encountered
while read t_empcode t_dept t_cl t_ml t_pl t_lwp t_da t_hra t_ca
    t_cca t_sppay_1 t_sppay_2 t_gs t_gpf t_gis t_esis t_inc_tax
    t_prof_tax t_rent_ded t_lt_loan t_st_loan t_spded_1 t_spded_2
    t_tot_ded t_net_pay
do
    if [ "$empcode" = "$t_empcode" ]
    then
        break
    fi
done

# reset standard input to terminal
exec < $t

IFS="$IFSspace"

# display various field values at appropriate places
writerc "Employee code:" 3 1 "B"
writerc "$e_empcode" 3 15 "N"
writerc "\033[1mSex:\033[0m$e_sex" 3 24 "N"
```

```
writerc "\033[1mGrade:\033[0m$e_grade" 3 40 "N"
writerc "\033[1mMonth:\033[0m$month" 3 66 "N"
writerc "\033[1mName:\033[0m$e_empname" 5 1 "N"
writerc "\033[1mDepartment:\033[0m$e_dept" 5 50 "N"
writerc "\033[1mGPF NO.:\033[0m$e_gpf_no" 7 1 "N"
writerc "\033[1mGIS NO.:\033[0m$e_gis_no" 7 25 "N"
writerc "\033[1mESIS NO.:\033[0m$e_esis_no" 7 48 "N"
writerc "Normal Days" 9 1  "B"
writerc "Casu.Leav" 9 21 "B"
writerc "Medical Leave" 9 32 "B"
writerc "Prov. Leave" 9 48 "B"
writerc "LWP" 9 61 "B"
writerc "Attended Days" 9 66 "B"
writerc "$mdays" 10 1 "N"
writerc "$t_cl" 10 21  "N"
writerc "$t_ml" 10 32 "N"
writerc "$t_pl" 10 48 "N"
writerc "$t_lwp" 10 61 "N"
writerc "$e_cum_att" 10 66 "N"
writerc "BS" 12 1 "B"
writerc "DA" 12 10 "B"
writerc "HRA" 12 25 "B"
writerc "CA" 12 32 "B"
writerc "CCA" 12 39 "B"
writerc "S.P.1" 12 48 "B"
writerc "S.P.2" 12 54 "B"
writerc "GS" 12 61 "B"
writerc "$e_bs" 13 1 "N"
writerc "$t_da" 13 10 "N"
writerc "$t_hra" 13 25 "N"
writerc "$t_ca" 13 32 "N"
writerc "$t_cca" 13 39 "N"
writerc "$t_sppay_1" 13 48 "N"
writerc "$t_sppay_2" 13 54 "N"
writerc "$t_gs" 13 61 "N"
writerc "GPF" 15 1 "B"
writerc "ESIS" 15 10 "B"
```



```
    writerc "GIS" 15 18 "B"
    writerc "IT" 15 25 "B"
    writerc "PT" 15 32 "B"
    writerc "RENT" 15 39 "B"
    writerc "Loan1" 15 48 "B"
    writerc "Loan2" 15 54 "B"
    writerc "S.D.1" 15 61 "B"
    writerc "S.D.2" 15 68 "B"
    writerc "Total" 15 74 "B"
    writerc "$t_gpf" 16 1 "N"
    writerc "$t_esis" 16 10 "N"
    writerc "$t_gis" 16 18 "N"
    writerc "$t_inc_tax" 16 25 "N"
    writerc "$t_prof_tax" 16 32 "N"
    writerc "$t_rent_ded" 16 39 "N"
    writerc "$t_lt_loan" 16 48 "N"
    writerc "$t_st_loan" 16 54 "N"
    writerc "$t_spded_1" 16 61 "N"
    writerc "$t_spded_2" 16 68 "N"
    writerc "$t_tot_ded" 16 73 "N"
    writerc "Net Pay" 18 1 "B"
    writerc "Rs. $t_net_pay" 19 1 "N"
    writerc "Receiver's Signature" 19 59 "B"
    writerc "$dln" 20 1 "B"

    writerc "Want to display another payslip y/n " 22 10 "N"
    read another
done
```

spaysheet.prg

```
# spaysheet.prg
# Prints department-wise summary payroll sheet.

clear
```

```
# possible departments in the company
dept="MFG:ASSLY:STORES:MAINT:ACCTS"

t='tty'
IFSspace="$IFS"
IFScolon=":"

month='date '+%B''
year='date '+%Y''

# display report titles
writecentre "Payroll Processing System" 1 "B"
writecentre "Summary Payroll Sheet" 2 "B"
writecentre  "$month $year" 3 "B"

# display column headings
writerc "Total" 5 20 "B"
writerc "Gross" 5 35 "B"
writerc "Gross" 5 50 "B"
writerc "Net" 5 70 "B"
writerc "Department" 6 5 "B"
writerc "Employees" 6 20 "B"
writerc "Earning" 6 35 "B"
writerc "Deduction" 6 50 "B"
writerc "Payments" 6 70 "B"

count=1
row=8

# run the loop for 5 different departments in the company
while [ $count -le 5 ]
do
    # pick up one department
    var='echo $dept | cut -d":" -f $count'

    # initialise variables
    tot_emp=0
```



```
gross_earn=0
gross_ded=0
net_pay=0
IFS="$IFScolon"

# set standard input to transaction file
exec < $TRAN

# read records from transaction file
while read t_empcode t_dept t_cl t_ml t_pl t_lwp t_da t_hra t_ca
      t_cca t_sppay_1 t_sppay_2 t_gs t_gpf t_gis t_esis t_inc_tax
      t_prof_tax t_rent_ded t_lt_loan t_st_loan t_spded_1 t_spded_2
      t_tot_ded t_net_pay
do
      IFS="$IFSspace"

      # if department matches
      if [ "$t_dept" = "$var" ]
      then
            tot_emp='expr $tot_emp + 1'
            gross_earn='echo "scale = 2\n $gross_earn + $t_gs" | bc'
            gross_ded='echo "scale = 2 \n $gross_ded + $t_tot_ded" | bc'
            net_pay='echo "scale = 2 \n $net_pay + $t_net_pay" | bc'
      fi

      IFS="$IFScolon"
done

# reset standard input to terminal
exec < $t

# output summary values of one department
writerc "$var" $row 5 "N"
writerc "$tot_emp" $row 20 "N"
writerc "$gross_earn" $row 35 "N"
writerc "$gross_ded" $row 50 "N"
writerc "$net_pay" $row 70 "N"
```

```
    IFS="$IFSspace"
    row=`expr $row + 1`
    count=`expr $count + 1`
done

IFS="$IFSspace"

writerc "Press any key..." 24 10 "N"
read key
```

lsr.prg

```
# lsr.prg
# Generates leave status report.

clear

# initialise variables
another=y
t=`tty`
month=`date '+%B'`
IFSspace="$IFS"

while [ "$another" = y  -o  "$another" = Y ]
do
    clear
    writecentre "Payroll Processing System" 1 "B"
    writecentre "Leave Status Report" 2 "B"

    writerc "Employee Code: " 4 10 "B"
    read empcode

    if [ -z "$empcode" ]
    then
        exit
```



```
    fi

    grep \^$empcode: $MASTER  >  /dev/null
    if [ $? -ne 0 ]
    then
        writerc "Employee code does not exist. Press any key..." 10 10 "B"
        read key
        continue
    fi

    IFS=":"

    # set standard input to master file
    exec  <  $MASTER

    # search for the desired employee code
    while read e_empcode e_empname e_sex e_address e_city e_pin
          e_dept e_grade e_gpf_no e_gis_no e_esis_no e_max_cl e_max_pl
          e_max_ml e_bs e_cum_cl e_cum_pl e_cum_ml e_cum_lwp
          e_cum_att
    do
        if [ "$empcode" = "$e_empcode" ]
        then
            break
        fi
    done

    # reset standard input to terminal
    exec  <  $t

    IFS="$IFSspace"

    # calculate balance leaves
    bal_cl=`expr $e_max_cl - $e_cum_cl`
    bal_ml=`expr $e_max_ml - $e_cum_cl`
    bal_pl=`expr $e_max_pl - $e_cum_pl`
```

```
# display leave status

writerc "                              " 4 1 "N"
writerc "\033[1mName:\033[0m$e_empname"  5 1 "N"
writerc "\033[1mEmpcode:\033[0m$e_empcode" 5 35 "N"
writerc "\033[1mGrade:\033[0m$e_grade" 5 55 "N"
writerc "\033[1mMonth:\033[0m$month" 5 66 "N"

writerc "CL Allowed" 7  1 "B"
writerc "ML Allowed" 7 15 "B"
writerc "PL Allowed" 7 35 "B"
writerc "$e_max_cl" 8 1 "N"
writerc "$e_max_ml" 8 15 "N"
writerc "$e_max_pl" 8 35 "N"

writerc "Cum.Cl" 10 1 "B"
writerc "Cum.ML" 10 15 "B"
writerc "Cum.PL" 10 35 "B"
writerc "Cum.LWP" 10 55 "B"
writerc "Cum.Att.Days" 10 66 "B"
writerc "$e_cum_cl" 11 1 "N"
writerc "$e_cum_ml" 11 15 "N"
writerc "$e_cum_pl" 11 35 "N"
writerc "$e_cum_lwp" 11 55 "N"
writerc "$e_cum_att" 11 66 "N"

writerc "Balance CL" 13 1 "B"
writerc "Balance ML" 13 15 "B"
writerc "Balance PL" 13 35 "B"
writerc "$bal_cl" 14 1 "N"
writerc "$bal_ml" 14 15 "N"
writerc "$bal_pl" 14 35 "N"

writerc "Another employee y/n " 21 10 "N"
read another
done
```


clmonth.prg

```
# clmonth.prg
# Closes the monthly transaction file.

clear
writecentre "Payroll Processing System" 2 "B"
writecentre "Close Current Month" 3 "B"

set 'date'
cur_mth=$2

if [ -f  "etran$cur_mth.dbf" ]
then
     writecentre "Month has already been closed. Press any key..." 15 "B"
 read key
 exit
 fi

 count='wc -l $MASTER'
 set $count
 mcount=$1
 count='wc -l $TRAN'
 set $count
 tcount=$1

 # if all records have not been entered in transaction file
 if [ $mcount -gt $tcount ]
 then
      writecentre "Transaction file incomplete. Cannot close month." 15 "B"
      writecentre "Press any key..." 16 "B"
      read key
      exit
 fi

 mv  $TRAN  etran$cur_mth.dbf
```

```
touch $TRAN

# check for success
if [ $? -eq 0 ]
then
     writecentre "Month successfully closed... Press any key" 15 "B"
else
     writecentre "Unable to close month... Press any key" 15 "B"
fi
read key
```

clyear.prg

```
# clyear.prg
# Closes the yearly transactions, updates master and reorganizes.

clear
writecentre "Payroll Processing System" 2 "B"
writecentre "Close Year & Reorganize" 3 "B"

t=`tty`
oldifs="$IFS"

writecentre "Please wait... trying to close year" 10 "B"

yr=`date '+%y'`
if [ -f "etran$yr.dbf" ]
then
   writecentre "Year has already been closed. Press any key..." 12 "B"
   read key
   exit
fi

# since financial year is from Apr to Mar
months="Apr:May:Jun:Jul:Aug:Sep:Oct:Nov:Dec:Jan:Feb:Mar"
IFS=:
```



```
set $months
count=1

flag=0
while [ $count -le 12 ]
do
    if [ -f  "etran$1.dbf" ]
    then
         cat  etran$1.dbf  >>  etran$yr.dbf
         rm  etran$1.dbf
         flag=1
    fi

    count=`expr $count + 1`
    shift
done

if [ $flag = 0 ]
then
   writecentre "Month has not been closed. Press any key..." 12 "B"
   writecentre "Close month before closing year. Press any key..." 13
"B"
   read key
   exit
fi

# set standard input to master file
exec  <  $MASTER

# prepare master file for new financial year
while read e_empcode e_empname e_sex e_address e_city e_pin
      e_dept e_grade e_gpf_no e_gis_no e_esis_no e_max_cl e_max_pl
      e_max_ml e_bs e_cum_cl e_cum_pl e_cum_ml e_cum_lwp
      e_cum_att
do
      e_cum_cl=0
```

```
        e_cum_pl=0
        e_cum_ml=0
        e_cum_lwp=0
        e_cum_att=0
        IFS="$oldifs"
        echo $e_empcode:$e_empname:$e_sex:$e_address:$e_city:
        $e_pin:$e_dept:$e_grade:$e_gpf_no:$e_gis_no:$e_esis_no:
        $e_max_cl:$e_max_pl:$e_max_ml:$e_bs:$e_cum_cl:$e_cum_pl:
        $e_cum_ml:$e_cum_lwp:$e_cum_att >> /tmp/master
        IFS=":"
    done

    IFS="$oldifs"
    mv /tmp/master $MASTER

    # reset standard input to terminal
    exec < $t
    writecentre "Year has been closed successfully. Press any key..." 12
    "B"
    read key
```

Where Do You Go from Here...

Phew! That was one long program listing. I Hope you understood the underlying logic. I can appreciate that such big programs cannot be imbibed to the last detail at first shot. But if you take it apart file by file the whole process of understanding would become a little easier. Now you can think of developing on your own the following programs which have been left as an exercise to the reader.

(a) tmodi.prg - Modification of an existing record in the transaction file.
(b) tdel.prg - Deletion of an existing record in the transaction file.
(c) tret.prg - Retrieval of an existing record from the transaction file.

Improve This Program...

However good one does anything there is always a scope for improvement. So also is true with this program.

You can improve this program in several ways. Some of these are mentioned below:

(a) The program is not sand-papered with error checks. (This was done to simply keep the programs compact and easy to understand.) For example, while performing data entry the program doesn't do data validation. That is, if the user supplies Basic salary as alphanumeric or Name as numeric the program blindly accepts this. The data validation can be done for each field. Though this is not entirely impossible using the shell techniques that we know, a better idea would possibly to call a C routine which can do the data validation more efficiently.

(b) The program has been hard-coded to implement only five different types of departments present in the company. Also, the program assumes there are certain fixed grades of employees. You can improve upon this by making the program work for any number of departments and grades of employees. It would be a good idea to read the percentages used for calculation of various allowances and deductions from an initialisation file when the program is run every time rather than hard-coding these facts within a program.

(c) No facility has been provided to take backup of current master and transaction files. This can be implemented through the System Maintenance menu.

(d) One more menu called File Tools can be implemented which would permit the usual file operations like copying, deletion, catenation, renaming etc. such that the user doesn't have to quit from the software just to perform these common chores.