ASSIGNMENT 11

Aim: Department maintain a student information. The file contains roll no., name, division and address. Write a program to create a sequential file to store and maintain student data. It should allow the user to add, delete information of student. Display information of particular employee. If record of student does not exist an appropriate message is displayed. If student record is found it should display student details.

Implement sequential file for student data base and perform operations on it:

i) Create DB
ii) Display DB
iii) Add record
iv) Delete Record.

Theory :

File : A file is a collection on information, usually stored on a computer's disk. Information can be saved to files and then reused later.

Type of file :

- Binary file :
  - The binary file consists of binary data
  - It can store texts, graphics, sound data in binary format
  - The binary files cannot be read directly
  - Numbers stored efficiently.

· Text file

- The text file contains the plain ASCII characters
- It contains text data which is marked by 'end of line' at the end of each record.
- Text file cannot store graphical data.
- This end of record marks help easily to perform operations such as read and write.

File organisation:

The proper arrangement of records within a file is called as file organisation. The factors that affect file organisation ae mainly the following:

- storage device
- Type of query
- Number of Keys.
- Mode of retrieval /update of record.

Diffeent types of file organisation ae:

- Sequential file
- Direct or random acess file
- Indexed sequential file.
- Multi indexed file.

Sequential file: Records are stored in ~~se~~ sequential order of their entry. This is the simplest kind of data organisation. The order of records is fixed. Within each block, the records are in sequence. New records always appear at end of the file.

Features of sequential files:

- Records stored in pre-defined order
- sequential access to successive records
- Suited to magnetic tape
- Very useful for transaction processing where the hit rate is very high.

Drawbacks of sequential files:

- Insertion and deletion of records in-between positions huge data movement.
- Acessing any record requires a pass through all the preceding records, which is time consuming.
- Needs reorganisation of file from time to time.

Primitive operations:

Open: Opens file and sets pointer to immediately before first record

Read-next: Returns next record to user.

Close: Closes file terminates access to file.

write-next: Pointers are set to next of last record and write record to file.

```
EOF:     if end of file
                 return tre;
           return false;
```

Serch: Search for the record with a gicn key

Update: Curent record is ~~wit~~ writlen at sare position with updated vles.

Algorithm:

(i) main():

```
- Read 2 filenames from user master and temporary
- Read operations to be performed
- if operation == "create"
          create();
  else if operation == "display"
          display();
  else if operation == "delete"
          delete();
  else if operation == "serch"
          serach();
  else if operation == "modify"
          modify();
  else
      exit();
```

(ii) create() :

- Open file in write mode
- Read no of records to be inserted in file.
- Read the no. of records one by one from each student and write in the file.
- Close file
- Return to main()

(iii) display()

- Open file in read mode.
- if file is present go to next step else go to previous step with error message.
- Scan all student records one by one and display
- Close file.
- Return to main()

(iv) add() :

- Open file in append mode
- Scan student record and write them in file at end
- Close file
- Return to main() :

(v) delete():

```
- Open file in append mode.
- Accept roll no of student record to delete
- Serch for roll no. in file.
      if file exists
          copy all records in file except the one
          to be deleted in a temporary file
      end if
- close both files
- Rename temporary file and remove old file.
- Return to main():
```

(vi) serch():

```
- Open the file in read mode
- Read roll no. of student record to search
- Read each record in file
- Compare roll numbers.
      if equal
          Display student record
      else
          Display "Record not found"
- Close file.
- Return to main().
```

Conclusion : we have implemented sequential file and performed all primitive operations on it.