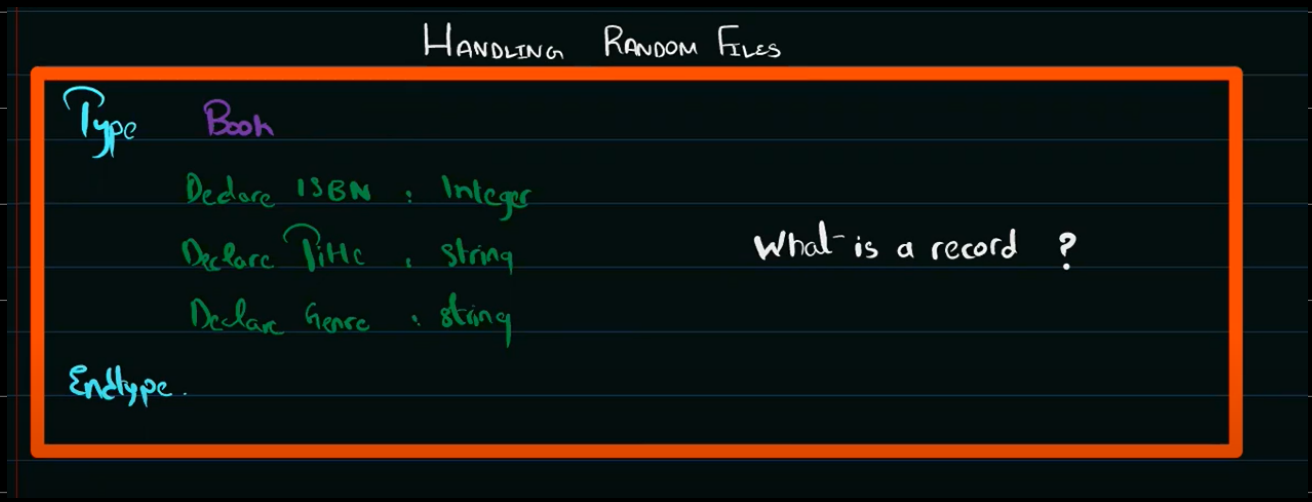
Files And Exception Handling

Note: Revise Lectures of Files from AS



Files

Binary Files:

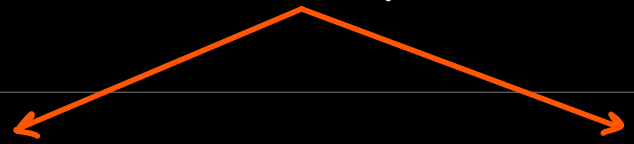
- .dat extension
- Data is stored in binary format directly
- Can be accessed directly

Text Files

- .txt file extension
- accessed sequentially

Random Files

* File organization & Access



Read / Write at the same time

Data can be accessed directly

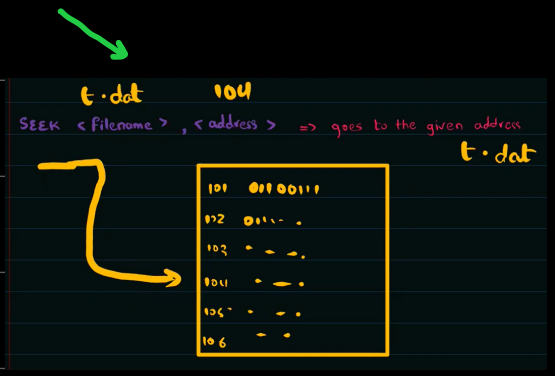
- Random file contains a collection of data
- Normally as records of fixed length
- They can be thought of as having a file pointer which can be moved to any address or location in a file.
- The record at that location can then be read or written

Pseudocode

- OPEN <filename> FOR RANDOM ; To open a random file

The SEEK command moves the file pointer to the given address

- SEEK <filename> , <address> ; goes to the given address



The command GETRECORD should be used to read the record at the file pointer.

- GETRECORD <filename> , <identifier> ; read the particular record

When this command is executed, the variable is assigned to the record that is read.

The command PUTRECORD is used to write a record into the file at the file pointer

- PUTRECORD <filename> , <identifier> ; put the particular record

Example

```
TYPE Book
    DECLARE BookID : INTEGER
    DECLARE BookName: STRING
    DECLARE Author: STRING
END TYPE

DECLARE Novel: Book
Novel.BookID ← 235
Novel.BookName ← "Goosebumps"
Novel.Author ← "R.L stine"

OPENFILE "Papersdock.dat" FOR RANDOM
Address ← Hash ( Novel.BookID)
SEEK "Papersdock.dat" , Address
```

```
PUTRECORD "Papersdock.dat", Novel
CLOSEFILE "Papersdock.dat"
```

Q- Imagine there is a file with 10 records "Random.dat". Ask user bookID and print the book name.

```
OPENFILE "Random.dat" FOR RANDOM
OUTPUT "Enter BookID"
INPUT BookID
Address ← Hash(BookID)
SEEK "Random.dat", Address
GETRECORD "Random.dat", Book
OUTPUT "Book name: ", Book.BookName
CLOSEFILE "Random.dat"
```

Q- The records from positions 10 to 20 of a file studentfile.dat are moved to the next position and a new record is inserted into position 10. Assume new record is already given.

```
OPENFILE "Studentfile.dat" FOR RANDOM
FOR Position ← 20 TO 10 STEP -1
    SEEK "studentfile.dat", Position
    GETRECORD "Studentfile.dat", Data
    SEEK "Studentfile.dat", Position +1
    PUTRECORD "studentfile.dat", Data
END FOR
SEEK "studentfile.dat", 10
PUTRECORD "Studentfile.dat", NewRecord
CLOSEFILE "Studentfile.dat"
```

Exception Handling

Q- What is meant by exception?

- It is an unplanned event and a situation causing crash

Q- What is meant by exception Handling?

- Code which is called when a run-time error occurs
- to avoid program crashing

Q- What are the situations where an exception handling routine would be required?

- Division by zero
- Run-time error
- File does not exist
- Invalid array index
- Invalid input
- Hardware Failure

Q- Describe the benefits of using exception handling in a program

- The program will not crash
- Results does not cause further errors
- Appropriate error message can be displayed
- Exceptional conditions are identified
- Improves readability

(c) A program is to be written to read a list of exam marks from an existing text file into a 1D array.

Each line of the file stores the mark for one student.

State **three** exceptions that a programmer should anticipate for this program.

1 Open a non existing text file

2 File already open in a different mode

3 Non-integer value read

[3]

The structure for error handling can be shown in pseudocode as:

```
TRY
    <statements>
EXCEPT
    <statements>
ENDTRY
```