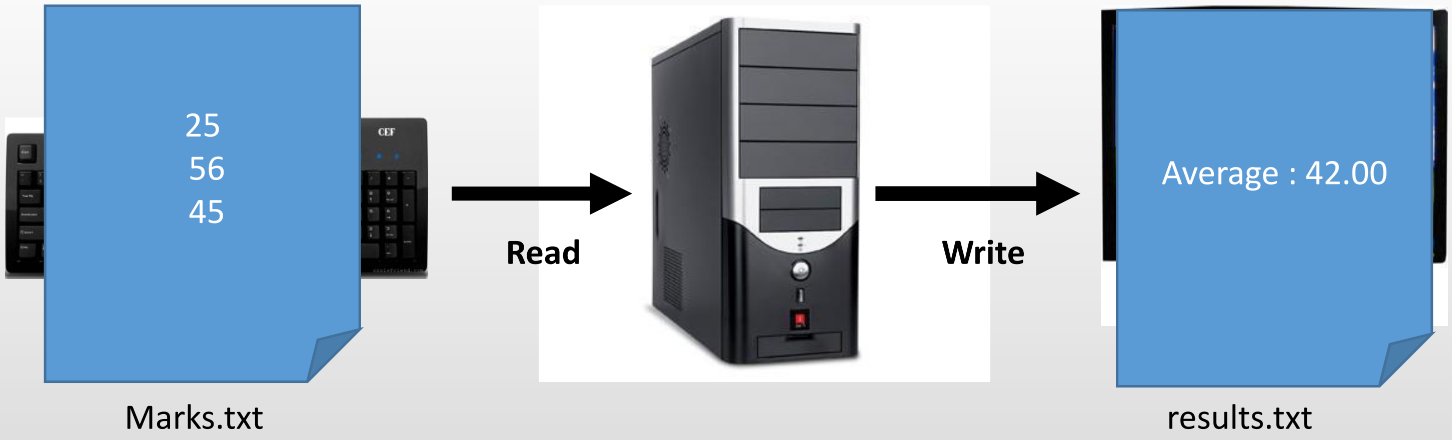
File Handling in C

Objectives

- At the end of the Lecture students should be able to
  - Create, update and process data files for storing and reading data.

Sequential Files

- Storage of data in variables and arrays is temporary.
- Data is lost when program terminates.
- Files are used to store data permanently.



25
56
45

Marks.txt

Read

Write

Average : 42.00

results.txt

Creating Sequential Access Files

Declaring a file pointer

```
FILE *cfPtr;
```

Open a file to write data.

```
cfPtr = fopen(“number.dat”, “w”);
```

Creates “number.dat” file to store/write data

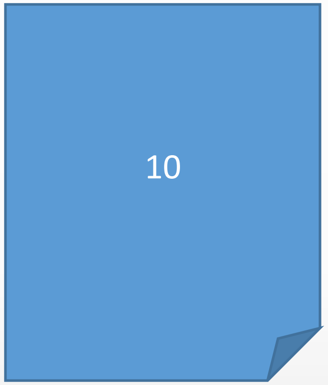
Writing data to a sequential-access file

```
#include <stdio.h>
int main(void)
{
     int number = 10;

     FILE *cfPtr;
     cftr = fopen(“number.dat”, “w”);

     if ( cfPtr == NULL)
     {
         printf(“Cannot create file\n”);
         return -1;
      }
      fprintf(cfPtr, “%d\n”, number);
      fclose(cfPtr);
      return 0;
}
```

data.dat



Close each file as soon as it’s no longer needed.

Exercise

- Write a program to input the id, name and average marks of a student from the keyboard and write the data to “marks.dat” file.

Writing multiple records to a sequential file

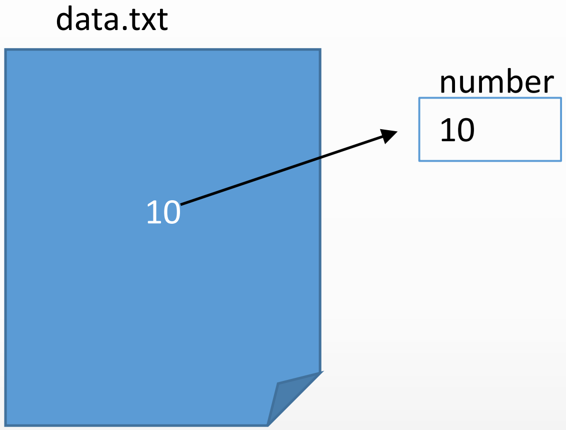
```
int main(void)
{
        char ID[10];
        char name[ 30];
        double avgMarks;
        int i;
        FILE *cfPtr;
        cfPtr = fopen(“marks.dat", "w");
        if (cfPtr == NULL)
        {
                printf("File cannot be open");
                return -1;
        }
        for(i = 1; i <= 5; ++i)
        {
                printf("Pls input the student ID");
                scanf("%s”, ID);
                printf("Pls input the name");
                scanf("%s", name);
                printf("Pls input the average Marks");
                scanf("%lf",& avgMarks);
                fprintf(cfPtr, "%s %s %.2f\n", ID, name, avgMarks);
        }
        fclose(cfPtr);
        return 0;
}
```

Reading data from a sequential – Access file

```
#include <stdio.h>
int main( void)
{
    int number ;

    FILE *cfPtr;
    cftr = fopen(“number.dat”, “r”);

    if ( cfPtr == NULL)
    {
         printf(“File could not be pened\n”);
         return -1;
   }
   fscanf(cfPtr, “%d”, &number);
   printf(“Number is : %d \n”, number );
   fclose(cfPtr);
    return 0;
}
```



Reading data from a file

```
# include <stdio.h>
int main(void)
{
        FILE *cfPtr;
         char ID[10];
        char name[ 30];
        double avgMarks;
        cfPtr = fopen(“marks.dat", “r");
        if ( cfPtr == NULL)
        {
                printf("File cannot be open");
                return -1;
        }
        fscanf(cfPtr, "%s %s %lf", ID, name, &avgMarks);
        printf "%s %s %lf", ID, name, avgMarks);
        fclose(cfPtr);
        return 0;
}
```

Reading multiple records from a sequential file

```
 int main(void)
{
          FILE *cfPtr;
           char ID[10];
          char name[ 30];
          double avgMarks;
          cfPtr = fopen(“marks.dat", “r");
          if ( cfPtr == NULL)
          {
                    printf("File cannot be open");
                    return -1;
          }
          fscanf(cfPtr, "%s %s %lf", ID, name, avgMarks);
          while (!feof(cfPtr))
          {
                    printf ("%s %s %lf", ID, name, avgMarks);
                    fscanf(cfPtr, "%s %s %lf", ID, name, &avgMarks);
          }
          fclose(cfPtr);
          return 0;
}
```

File Opening Modes

Mode	Description
r	Open an existing file for reading
w	Create a file for writing. If the file already exists, discard the current contents
a	Append; open or create a file for writing at the end of the file
r+	Open an existing fil for update (reading and writing)
w+	Create a file for update. If the file already exists, discard the current contents
a+	Append: open or create a file for update; writing is done at the end of the file.