File IO Part III

fread, fwrite, feof

fwrite and fread

- fread and fwrite are binary file reading and writing functions
  - Prototypes are found in stdio.h
- Generic Form:

```
int fwrite (void *buf, int size, int count, FILE *fp) ;
int fread  (void *buf, int size, int count, FILE *fp) ;
```

  - buf: is a pointer to the region in memory to be written/read
    - It can be a pointer to anything (more on this later)
  - size: the size in bytes of each individual data item
  - count: the number of data items to be written/read
- For example a 100 element array of integers
  - fwrite(buf, sizeof(int), 100, fp);
- The fwrite & fread returns the number of items actually written (read).

fwrite and fread

- Testing for errors:

```
if ((frwrite(buf,size,count,fp)) != count)
   fprintf(stderr, "Error writing to file.");
```

- Writing a single double variable x to a file:

```
fwrite (&x, sizeof(double), 1, fp) ;
```

  - This writes the double x to the file in raw binary format
    - i.e., it simply writes the internal machine format of x
- Writing an array text[50] of 50 characters can be done by:
  - `fwrite (text, sizeof(char), 50, fp) ;`

    or
  - `fwrite (text, sizeof(text), 1, fp);  /* text must be a local array name */`
- fread and frwrite are more efficient than fscanf and fprintf

A Sample program to show fread & fwrite

```c
#include <stdio.h>
#include <string.h>

int main()
{
     int numb=50;
     char Name[]="sarava";
     FILE *ptr=fopen("ha.dat","w");
     fwrite(Name,sizeof(char),strlen(Name)+1,ptr);
     fwrite(&numb,sizeof(int),1,ptr);
     fclose(ptr);

     int tempnumb;
     char tempName[7];
     ptr=fopen("ha.dat","r");
     fread(tempName,sizeof(char),strlen(Name)+1,ptr);
     fread(&tempnumb,sizeof(int),1,ptr);
     fclose(ptr);

     printf("tempName = %s , tempnumb = %d",tempName,tempnumb);

     return 0;
}
```

program output

```
tempName = sarava , tempnumb = 50
```

Program to show the use of fread & fwrite with structures

```
// C program for writing
// struct to file
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
// a struct to read and write
struct person
{
      int id;
      char fname[20];
      char lname[20];
};
```

Cont..

```
int main ()
{
      FILE *outfile, *infile;
      int f=0,d=0;
      // open file for writing
      outfile = fopen ("person.dat", "w");
      if (outfile == NULL)
      {
            fprintf(stderr,"\nError in file\n");
            exit (1);
      }
      struct person input1 = {1, "Rohit", "Sharma"};
      struct person input2 = {2, "Mahindra“,“Dhoni"};
```

Cont….

```
      // write struct to file
      f=fwrite (&input1, sizeof(struct person), 1, outfile);
      d=fwrite (&input2, sizeof(struct person), 1, outfile);
      if((f!= 0) && (d!= 0))
            printf("contents to file written successfully !\n");
      else
            printf("error writing file !\n");

// closing file
      fclose (outfile);
      struct person input;
// Open person.dat for reading
    infile = fopen ("person.dat", "r");
```

Cont….

```
        if (infile == NULL)
       {
              fprintf(stderr, "\nError opening file\n");
              exit (1);
       }
 // read file contents till end of file
    while(fread(&input, sizeof(struct person), 1, infile))
     printf ("id = %d name = %s %s\n", input.id, input.fname,
              input.lname);
// close file
    fclose (infile);
    return 0;
}
```

feof function

The *feof* function can be used to test for an end of file condition.

feof returns : A non-zero value is returned in the case that the end-of-file indicator associated with the FILE is set. Otherwise, zero is returned.

```c
#include <stdio.h>

int main()
{
      FILE *ptr=fopen("ha.dat","w");
      fprintf(ptr,"hajsof");
      fclose(ptr);

      ptr=fopen("ha.dat","r");
      char a;
      while(1)
      {
            a=getc(ptr);
            if(feof(ptr))
            {
                  printf("End of file reached\n");
                  break;
            }
            printf("%c\n",a);
      }
      fclose(ptr);
      return 0;
}
```

program output

```
h
a
j
s
o
f
End of file reached
```