Files

Concept of a file, Opening and Closing files, file input / output functions (standard library input / output functions for text files)

FILE STRUCTURE:

- ✓ I/O functions available are similar to their console counterparts; scanf becomes fscanf, printf becomes fprintf, etc., These functions read and write from file streams.
- ✓ As an example, the file stream structure FILE defined in the header file stdio.h in DOS is shown below:

```
typedef    struct
{
    int level;  /*  fill/empty level of buffer  */
    usigned flasgs; /*  File status flags   */
    char fd;   /*  File descriptor (handle) */
    unsigned char hold; /*  Ungetc char if no buffer */
    int bsize;  /*  Buffer size   */
    unsigned char _FAR *buffer; /*  Data transfer buffer */
    unsigned char _FAR *curp;/*  Current active pointer */
    unsigned istemp; /*  Temporary file indicator */
    short token;  /*  Used for validity checking */
} FILE;   /*  This is the FILE Object *
```

NOTE: *FILE is defined as New Structure Data Type in the stdio.h file as shown in above method.*

End Of File:

- ✓ EOF is a macro defined as an int with a negative value. It is normally returned by functions that perform read operations to denote either an error or end of input.
- ✓ Input from a terminal never really "ends" (unless the device is disconnected), but it is useful to enter more than one "file" into a terminal, so a key sequence is reserved to indicate end of input.
- ✓ Cntrl+Z is the key in DOS to end or terminate the input values.
- ✓ Cntrl+D is the key in UNIX to end or terminate the input values.

Example: /* A program to Write a file and read a file */

```
#include<stdio.h>
main( )
{
    FILE *fp;
    char ch;
```

```
    fp=fopen("DATA1.txt","w");
    printf("Enter the Text:\n");
    printf("Use Ctrl+z to stop entry \n");
    while((scanf("%c",&ch))!=EOF)
    fprintf(fp, "%c",ch);
    fclose(fp);
    printf("\n");
    fp=fopen("DATA1.txt","r");
    printf("Entered Text is:\n");
    while((fscanf(fp,"%c",&ch))!=EOF)   printf("%c",ch);
    fclose(fp);
}
```

Output:

```
Enter the Text:
Use Ctrl+z to stop entry
Hi Raju...
      how r u... how is ur studies...:-)
^Z                          // Cntrl+Z key terminated the input values


Entered Text is:
Hi Raju...
      how r u... how is ur studies...:-)
```

File Handling Functions:

Reading Character from a file: fgetc( ) or getc( )

- ✓ fgetc( ) or getc( ) is a predefined file handling function which is used to read a single character from a existing file opened in read("r") mode by fopen( ), which is same as like getchar( ) function.

Syntax:

ch_var= fgetc(filepointer); ***(Or)*** *ch_var=getc(filepointer);*

Example:

char ch;

ch=fgetc(fp); (Or) ch=getc(fp);

Where 'ch' is a character variable to be written to the file.

Where 'fp' is a file pointer object.

- ✓ getc( ) or fgetc( ) gets the next character from the input file to which the file pointer fp points to. The function getc( ) will return an end-of-file(EOF) marker when the end of the file has been reached or it if encounters an error.
- ✓ On Success the function fputc( ) or putc( ) will return the value that it has written to the file, otherwise it returns EOF.

Example: /* A program to Read a file */

```
#include<stdio.h>
main( )
{
   FILE *fp;
   char ch;
   fp=fopen("DATA1.txt","r");
   /*  DATA1.txt is an already existing file with content  */
   printf("Text from file is:\n");
   while((ch=fgetc(fp))!=EOF)
    /* (Or)     while((ch=getc(fp))!=EOF)*/
   printf("%c",ch);
   fclose(fp);
}
```

Output:

```
Text from file is:
Hi Raju...      how r u... how is ur studies...:-)
```

Writing Or Printing Character in a file: fputc( ) or putc( )

✓ fputc( ) or putc( ) is a predefined file handling function which is used to print a single character in a new file opened in write("w") mode by fopen( ), which is same as like putchar( ) function.

Syntax:

fputc(ch_var , filepointer); (Or) putc(ch_var , filepointer);

Example: fputc(ch, fp); (Or) putc(ch, fp);

Where 'ch' is a character variable. & Where 'fp' is a file pointer object.

Example: /* A program to Write a Character in a file and read a Character from a file */

```
#include<stdio.h>
main( )
{
      FILE *fp;
```

```
        char ch;
        fp=fopen("DATA1.txt","w");
        printf("Enter the Text:\n");
        printf("Use Ctrl+z to stop entry \n");
        while((ch=getchar( ) )!=EOF)
          fputc(ch,fp);
           /* (Or)
          putc(ch,fp);  */
        fclose(fp);
        printf("\n");
        fp=fopen("DATA1.txt","r");
        printf("Entered Text is:\n");
        while((ch=fgetc(fp))!=EOF)
        putchar(ch);
        fclose(fp);
    }
```

Output:

```
Enter the Text:
Use Ctrl+z to stop entry
Hi ...
      how are you... how is your
studies...:-)
^Z                          // Cntrl+Z key terminated the input values

Entered Text is:
Hi ...
      how are you... how is your
studies...:-)
```

Reading String from a file: fgets( )

- ✓ fgets( ) is a predefined file handling function which is used to read a line of text from an existing file opened in read("r") mode by fopen( ), which is same as like gets( ) function.

Syntax:

*char *fgets(char *s, int n,FILE *fp);*

or

fgets(ch_var ,str_length, filepointer);

Example:

```
char ch[10];
fgets(ch ,20, fp);
```

Where 'ch' is a string variable to be written to the file.

Where 'fp' is a file pointer object.

Where 20 is the string length in a file.

- ✓ The function fgets( ) read character from the stream fp into the character array 'ch' until a newline character is read, or end-of-file is reached. It then appends the terminating null character after the last character read and returns 'ch' if end-of-file occurs before reading any character an error occurs during input fgets( ) returns NULL.

Writing Or Printing String in a file: fputs( )

- ✓ fputs( ) is a predefined file handling function which is used to print a string in a new file opened in write("w") mode by fopen( ), which is same as like puts( ) function.

Syntax:

*int fputs(const char *s, FILE *fp);*

or

fputs(ch_var , filepointer);

Example:

```
char ch[10]="gitam";
fputs(ch, fp);
```

Where 'ch' is a string variable.

Where 'fp' is a file pointer object.

- ✓ The function fputs( ) writes to the stream fp except the terminating null character of string s, it returns EOF if an error occurs during output otherwise it returns a non negative value.

Example: /* A program to Write and Read a string in a file */

```
#include<stdio.h>
main()
{
```

```
  char name[20];
  char name1[20];
  FILE *fp;  fp=fopen("dream.txt","w");
  puts("Enter any String\n");
  gets(name); fputs(name,fp);
  fclose(fp);
  fp=fopen("dream.txt","r");
  fgets(name1,10,fp);       /* Here '10' is nothing but String Length, i.e., upto
  how-  puts("String from file is:\n");         many characters u want to access
  from a file.  */
  puts(name1);
  fclose(fp);
}
```

Output:

```
Enter any String:
hi ram how r u
String from file is:
hi ram ho
```

Reading and Printing only integer value: getw() and putw():

- ✓ The getw( ) and putw( ) are predefined file handling integer-oriented functions. They are similar to the getc( ) and putc( ) functions and are used to read and write integer values.
- ✓ These functions would be useful when we deal with only integer data.

Syntax for getw( ):

integervariable=getw(filepointer);

Example:

```
int n;
n= getw(fp);
```

Syntax for putw( ):

putw(integervariable, filepointer);

Example:

```
putw(n,fp);
```

Example: /* A program to Write and Read only one Integer Value in a file */

```
#include<stdio.h>
      main()
      {
          int n,m;
          FILE *fp=fopen("num.txt","w");
          puts("Enter a number:");
```

```
        scanf("%d",&n);
        putw(n,fp);  /* printing only integer value  in a file  */
        fclose(fp);
        fp=fopen("num.txt","r");
        m=getw(fp);  /*  reading only integer value from a file  */
        printf("From File int val=%d",m);
    }
```

Output1:

```
Enter a number: 16
From File int val=16
```

Output2:

```
Enter a number: 25    35    45
From File int val=25      /*  here it take only one value, basing on program
requirement */
```

Output3:

```
Enter a number: 25.5
Frm File int val=25
```

Output4:

```
Enter a number: A
Frm File int val=28056   /* here it doesn't print ASCII value of 'A', it just print
some-garbage value  */
```

Example: /* A program to Write and Read more Integer Values in a file */

```
#include<stdio.h> main()
  {
    int n,m;
    FILE *fp=fopen("num.txt","w");
```

```
    puts("Press Cntl+Z to end the values");   puts("Enter
    the numbers:");   while(scanf("%d",&n)!=EOF)
    putw(n,fp);
    fclose(fp);
    fp=fopen("num.txt","r");
    puts("Entered values in file are:");
    while((m=getw(fp))!=EOF)
    printf("%d\n",m);
  }
```

Output:

```
Press Cntl+Z to end the values Enter the numbers:
  1 2 10 20 30 45
  ^Z
Entered values in file are:
  1
  2
  10
  20
  30
  45
```

/* File program to read a character file and encrypts it by replacing each alphabet by its next alphabet cyclically i.e., z is replaced by a. Non-alphabets in the file are retained as they are. Write the encrypted text into another file */

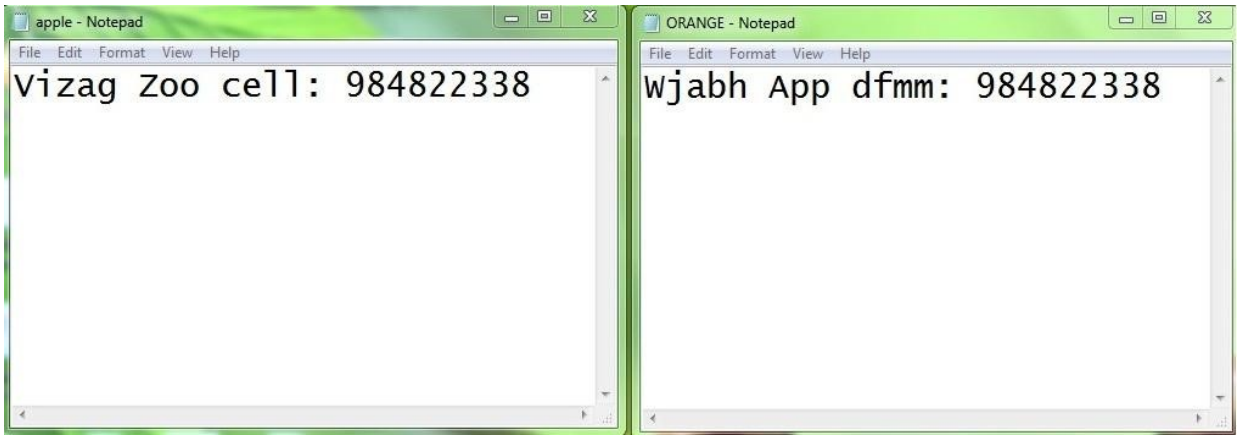
```
#include<stdio.h>
```

```
#include<stdlib.h>
 main()
 {
    FILE *fp1,*fp2;
    char fname1[20],fname2[20];
    char ch1,ch2;
    printf("Enter the source file name ");
    scanf("%s",fname1);
    fp1=fopen(fname1,"r");
    if(fp1==NULL)
    printf("%s is not available",fname1);
    else
       {
        printf("Enter the new file name ");
        scanf("%s",fname2);
        fp2=fopen(fname2,"w");
        while((ch1=fgetc(fp1))!=EOF)
          {
            printf("%c",ch1);
            if((ch1>=65 && ch1<=89) || (ch1>=97 && ch1<=121))
              ch2=ch1+1;
            else if(ch1==90 || ch1==122)
              ch2=ch1-25;
            else
               ch2=ch1;
            fputc(ch2,fp2);
           }
        printf("File copied into %s succesfull",fname2);
        fclose(fp1);
        fclose(fp2);
       }
 }
```

Output:

```
Enter the source file name apple.txt
Enter the new file name orange.txt
Vizag Zoo cell: 984822338File copied into orange.txt succesfull
```

Input text apple.txt Output text orange.txt

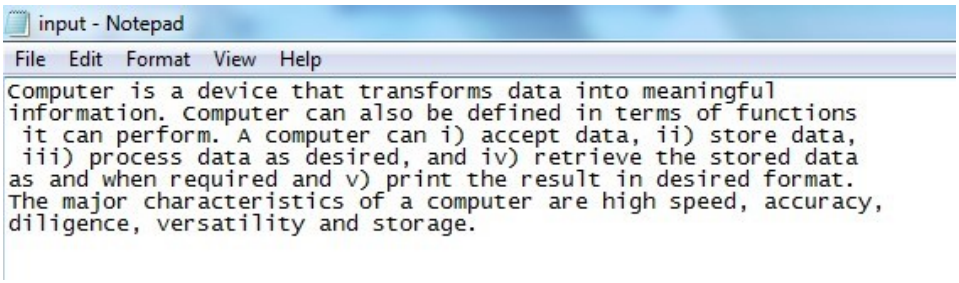


/*Program to read data from input file and place first 10 characters in an array and display on the monitor.*/

```
#include<stdio.h>
 main()
 {
  FILE *fp;
  int i=0;
  char c,a[11];
  fp=fopen("INPUT","r");
  while(i<100)
  {
  c=getc(fp);
  a[i]=c;
  i++;
  }
  fclose(fp);
  a[i]='\0';
  puts(a);
 }
```

Output:

```
 Computer i
```



/*Program to read a C program and count number of statement terminators and number of opening braces.*/

```
#include<stdio.h>
 main()
```

```
{
 FILE *fp;
 int s=0,b=0;
 char c;
 fp=fopen("fib.c","r");
  while((c=getc(fp))!=EOF)
  {
  if(c==';') s++;
  if(c=='{') b++;
  }
 fclose(fp);
 printf("\t\n number of statement terminators=%d",s);
 printf("\t\n no of opening braces=%d",b);
}
```

FIB - Notepad

File Edit Format View Help

```
#include<stdio.h>

void main() {
   int f1=0,f2=1,f,num,i=2;
   printf("Enter the number of terms : ");
   scanf("%d", &num);
   printf("\nFibonacci Series : %d  %d  ", f1, f2);

   if(num>2)
      {
      do
      {
      f = f1 + f2;
      printf("%d  ", f);
      f1 = f2;
      f2 = f;
      i++;
   }while(i<num);
   }
   }
```

Output:

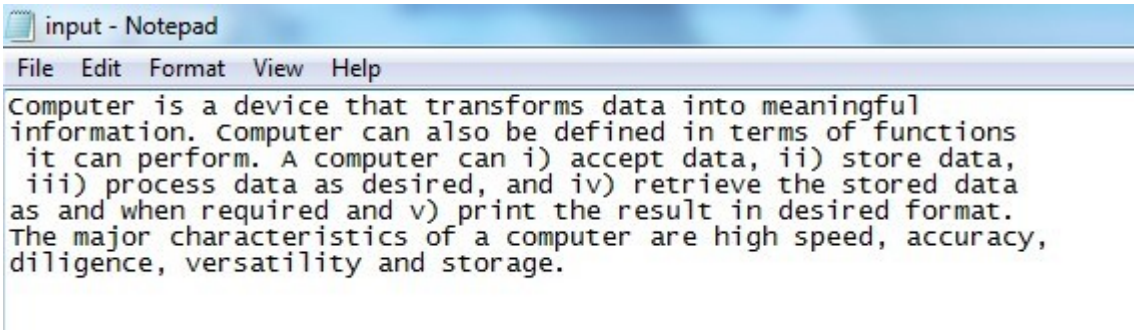
Number of Statements terminators=10

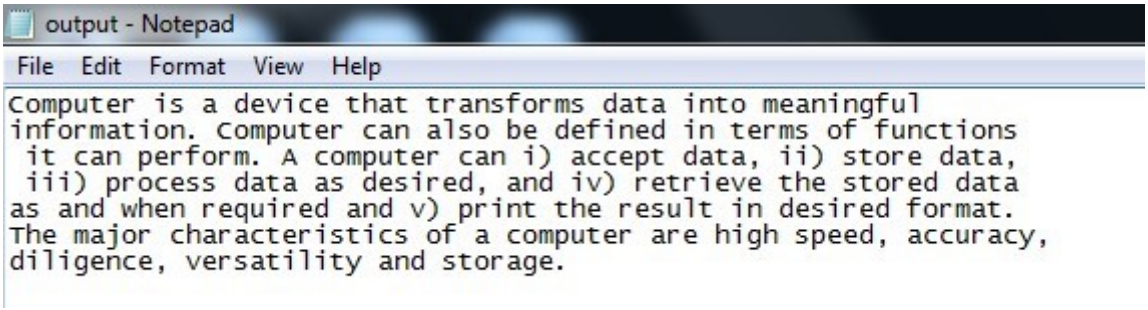
No.of Opening braces=3

/*Program to copy contents of one file into another file.*/

```
#include<stdio.h>
 main()
 {
```

```
    FILE *fp1,*fp2;
    char s;
    fp1=fopen("input.txt","r");
    fp2=fopen("output.txt","w");
    while ((s=getc(fp1))!=EOF)
    putc(s,fp2);
    fclose(fp1);
    fclose(fp2);
}
```

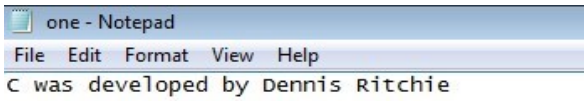


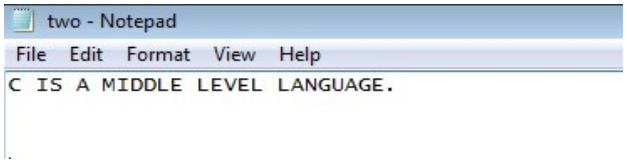


/*Program to append the contents to a file and display the contents before and after appending.*/

```
#include<stdio.h>
#include<conio.h>
main()
```

```
{
  FILE *fp1,*fp2;
  char s,c;
  clrscr();
  printf("\n\n\t one.txt contents are \n\n");
  /*prints contents of file1 on monitor*/
  fp1=fopen("one.txt","r");
  while((c=getc(fp1))!=EOF)
  printf("%c",c); fclose(fp1);
  printf("\n\n\t two.txt contents before appending are \n\n");
  /*prints contents of file2 on monitor before appending*/
  fp2=fopen("two.txt","r");
  while((c=getc(fp2))!=EOF)
  printf("%c",c); fclose(fp2);
  /*appends contents of file1 to file2*/
  fp1=fopen("one.txt","r");
  fp2=fopen("two.txt","a");
  while((c=getc(fp1))!=EOF)
  putc(c,fp2); fcloseall();
  printf("\n\n\t two.txt contents after appending are \n\n");
  /*prints contents of file2 on monitor after appending*/
  fp2=fopen("two.txt","r");
  while((c=getc(fp2))!=EOF)
  printf("%c",c);
  fclose(fp2);
}
```





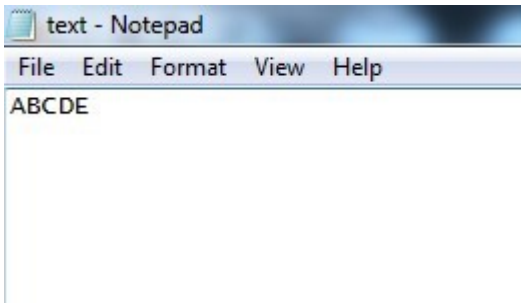
Output:

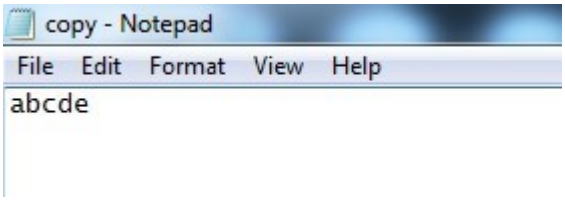
```
one.txt contents are
C was developed by Denis Ritchie two.txt contents before
appending are
C IS A MIDDLE LEVEL LANGUAGE.
two.txt contents after appending are
C IS A MIDDLE LEVEL LANGUAGE. C was developed by Denis Ritchie
```

/*Program to change all upper case letters in a file to lower case letters and vice versa.*/

```
#include<stdio.h>
 main()
```

```
{
  FILE *fp1,*fp2;
  char c;
  fp1=fopen("text.txt","r");
  fp2=fopen("copy.txt","w");
  while((c=getc(fp1))!=EOF)
  {
    if(c>=65&&c<=91)
    c=c+32;
    else
    c=c-32;
    putc(c,fp2);
  }
  fcloseall();
}
```





/*Program to read numbers from a file "data" which contains a series of integer numbers and then write all odd numbers to the file to be called "odd" and all even numbers to a file called "even".*/

```
#include<stdio.h>
main()
{
  FILE *fp,*fp1,*fp2;
```

```
  int c,i;
  clrscr();
  fp=fopen("data.txt","w");
  printf("enter the numbers");
  for(i=0;i<10;i++)
  {
    scanf("%d",&c);
    putw(c,fp);
  }
  fclose(fp);
  fp=fopen("data.txt","r");
  fp1=fopen("even.txt","w");
  fp2=fopen("odd.txt","w");
  while((c=getw(fp))!=EOF)
  {
    if(c%2==0)
    putw(c,fp1);
    else
    putw(c,fp2);
  }
  fclose(fp);
  fclose(fp1);
  fclose(fp2);
  fp1=fopen("even.txt","r");
  while((c=getw(fp1))!=EOF)
  printf("%4d",c);
  printf("\n\n");
  fp2=fopen("odd.txt","r");
  while((c=getw(fp2))!=EOF)
  printf("%4d",c);
  fcloseall();
}
```

Output:

```
Enter the numbers   1 2 3 4 5 6 7 8 9 10
Enter the numbers   1 2 3 4 5 6 7 8 9 10

2 4 6 8 10
 1 3 5 7 9
```