File Handling in C

Creating or opening file using fopen()

The fopen() function is used to create a new file or open an existing file in C. The fopen function is defined in the stdio.h header file.

Mode = “r” − open for reading, this mode will open the file for reading purpose only, i.e. the contents can only be viewed, nothing else like edits can be done to it.

This mode cannot create a new file and fopen() returns NULL, if we try to create a new file using this mode.

```
#include <stdio.h>
int main(){
  FILE * file;
  if (file = fopen("hello.txt", "r")){
        printf("File opened successfully in read mode");
  }
  else
  printf("The file is not present! cannot create a new file using r mode");
  fclose(file);
  return 0;
}
```

o/p
File opened successfully in read mode

Mode = “rb” − open for reading in binary mode, this mode will open the file for reading in binary mode only, i.e. the contents can only be viewed and nothing else like edits can be done to it.

This mode cannot create a new file and fopen() returns NULL, if we try to create a new file using this mode.

```
#include <stdio.h>
int main(){
  FILE * file;
  if (file = fopen("program.txt", "rb")){
        printf("File opened successfully in read mode");
  }
  else
  printf("The file is not present! cannot create a new file using rb mode");
```

```
  fclose(file);
  return 0;
}
```

Output
The file is not present! cannot create a new file using rb mode

Mode = "w" − open for writing only, this mode will open the file if present in the current directory for writing only i.e. reading operation cannot be performed. If the file is not present in the current directory, the program will create a new file and open it for writing.

If we open a file that contains some text in it, the contents will be overwritten.

```
#include <stdio.h>
int main(){
  FILE * file;
  if (file = fopen("helo.txt", "w")){
        printf("File opened successfully in write mode or a new file is created");
  }
  else
  printf("Error!");
  fclose(file);
  return 0;
}
```

Output
File opened successfully in write mode or a new file is created

Mode = "wb" − open for writing in binary mode, this mode will open the file if present in the current directory for writing in binary mode i.e. reading operation cannot be performed. If the file is not present in the current directory, the program will create a new file and open it for writing in binary mode.
If we open a file that contains some text in it, the contents will be overwritten.

```
#include <stdio.h>
int main(){
  FILE * file;
  if (file = fopen("hello.txt", "wb")){
        printf("File opened successfully in write in binary mode or a new file is created");
  }
  else
  printf("Error!");
```

```
  fclose(file);
  return 0;
}
```

Output
File opened successfully in write in binary mode or a new file is created

Mode = “a” − open for append only, this mode will open the file if present in the current directory for writing only i.e. reading operation cannot be performed. If the file is not present in the current directory, the program will create a new file and open it for writing. If we open a file that contains some text in it, the contents will not be overwritten; instead the new text will be added after the existing text in the file.

```
#include <stdio.h>
int main(){
  FILE * file;
  if (file = fopen("hello.txt", "a")){
        printf("File opened successfully in append mode or a new file is created");
  }
  else
  printf("Error!");
  fclose(file);
  return 0;
}
```

Output
File opened successfully in append mode or a new file is created

Mode = “ab” − open for append in binary, this mode will open the file if present in the current directory for writing in binary only i.e. reading operation cannot be performed. If the file is not present in the current directory, the program will create a new file and open it for writing in binary.

If we open a file that contains some text in it, the contents will not be overwritten; instead the new text will be added after the existing text in the file.

```
#include <stdio.h>
int main(){
  FILE * file;
  if (file = fopen("hello.txt", "ab")){
        printf("File opened successfully in append in binary mode or a new file is created");
  }
```

```
  else
  printf("Error!");
  fclose(file);
  return 0;
}
```

Output
File opened successfully in append in binary mode or a new file is created

Mode = "r+" − open for reading and writing both, this mode will open the file for both reading and writing purposes i.e. both read and write operations can be performed to the file.
This mode cannot create a new file and fopen() returns NULL, if we try to create a new file using this mode.
If we open a file that contains some text in it and write something, the contents will be overwritten.

```
#include <stdio.h>
int main(){
  FILE * file;
  if (file = fopen("hello.txt", "r+")){
        printf("File opened successfully in read and write both");
  }
  else
  printf("The file is not present! cannot create a new file using r+ mode");
  fclose(file);
  return 0;
}
```

Output
File opened successfully in read and write both

Mode = "rb+" − open for reading in binary mode, this mode will open the file for reading in binary mode only, i.e. the contents can only be viewed and nothing else like edits can be done to it.
This mode cannot create a new file and fopen() returns NULL, if we try to create a new file using this mode.
If we open a file that contains some text in it and write something, the contents will be overwritten.

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

```
  if (file = fopen("program.txt", "rb+")){
        printf("File opened successfully in read mode");
  }
  else
  printf("The file is not present! cannot create a new file using rb+ mode");
  fclose(file);
  return 0;
}
```

Output
The file is not present! cannot create a new file using rb+ mode

Mode = "w+" − open for writing and reading, this mode will open the file if present in the current directory for writing and reading operation both. If the file is not present in the current directory, the program will create a new file and open it for reading and writing.
If we open a file that contains some text in it, the contents will be overwritten.

```
#include <stdio.h>
int main(){
  FILE * file;
  if (file = fopen("helo.txt", "w+")){
        printf("File opened successfully in read-write mode or a new file is created");
  }
  else
  printf("Error!");
  fclose(file);
  return 0;
}
```

Output
File opened successfully in read-write mode or a new file is created

Mode = "wb+" : open for writing and reading in binary mode, this mode will open the file if present in the current directory for writing and reading in
binary mode. If the file is not present in the current directory, the program will create a new file and open it for reading and writing in binary mode. If we open a file that contains some text in it, the contents will be overwritten.

```
#include <stdio.h>
int main(){
  FILE * file;
  if (file = fopen("hello.txt", "wb+")){
```

```
        printf("File opened successfully in read-write in binary mode or a new file is created");
  }
  else
  printf("Error!");
  fclose(file);
  return 0;
}
```

Output
File opened successfully in read-write in binary mode or a new file is created

Mode = "a+" − open for read and append, this mode will open the file if present in the current directory for both reading and writing. If the file is not present in the current directory, the program will create a new file and open it for reading and writing.

If we open a file that contains some text in it, the contents will not be overwritten; instead the new text will be added after the existing text in the file.

```
#include <stdio.h>
int main(){
  FILE * file;
  if (file = fopen("hello.txt", "a+")){
        printf("File opened successfully in read-append mode or a new file is created");
  }
  else
  printf("Error!");
  fclose(file);
  return 0;
}
```

Output
File opened successfully in read-append mode or a new file is created

Mode = "ab+" − open for read and append in binary, this mode will open the file if present in the current directory for both reading and writing in binary. If the file is not present in the current directory, the program will create a new file and open it for reading and writing in binary. If we open a file that contains some text in it, the contents will not be overwritten; instead the new text will be added after the existing text in the file.

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

```
  if (file = fopen("hello.txt", "ab+")){
        printf("File opened successfully in read-append in binary mode or a new file is created");
  }
  else
  printf("Error!”);
  fclose(file);
  return 0;
}
```

Output

File opened successfully in read-append mode or a new file is created

Practice

```
#include <stdio.h>
int main(){
  FILE *fp;
  fp = fopen("file.txt", "w");//opening file
  fprintf(fp, "Hello file \n");//writing data into file
  fclose(fp);//closing file
return 0;
}
```

```
#include <stdio.h>
int main(){
  FILE *fp;
  char buff[255];//creating char array to store data of file
  fp = fopen("file.txt", "r");
  while(fscanf(fp, "%s", buff)!=EOF){
  printf("%s ", buff );
  }
  fclose(fp);
}
```

The fputc() function is used to write a single character into file.
It outputs a character to a stream.

```
#include <stdio.h>
int main(){
  FILE *fp;
  fp = fopen("file1.txt", "w");//opening file
  fputc('a',fp);//writing single character into file
  fclose(fp);//closing file
}
```

The fgetc() function returns a single character from the file.
It gets a character from the stream. It returns EOF at the end of file.

```
#include<stdio.h>
#include<conio.h>
void main(){
FILE *fp;
char c;
clrscr();
fp=fopen("myfile.txt","r");
while((c=fgetc(fp))!=EOF){
printf("%c",c);
}
fclose(fp);
getch();
}
```

The fputs() function writes a line of characters into file.
It outputs string to a stream.

```
#include<stdio.h>
#include<conio.h>
void main(){
FILE *fp;
clrscr();
fp=fopen("myfile2.txt","w");
fputs("hello c programming",fp);
fclose(fp);
getch();
}
```

The fgets() function reads a line of characters from file.
It gets string from a stream.

```
#include<stdio.h>
#include<conio.h>
void main(){
FILE *fp;
char text[300];
clrscr();
fp=fopen("myfile2.txt","r");
printf("%s",fgets(text,200,fp));

fclose(fp);
```

```
getch();
}
```