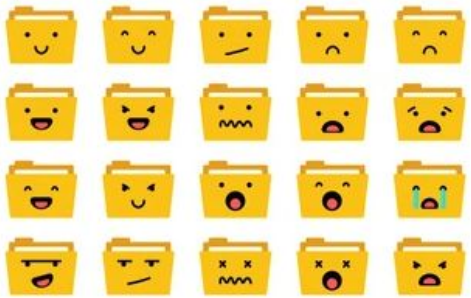
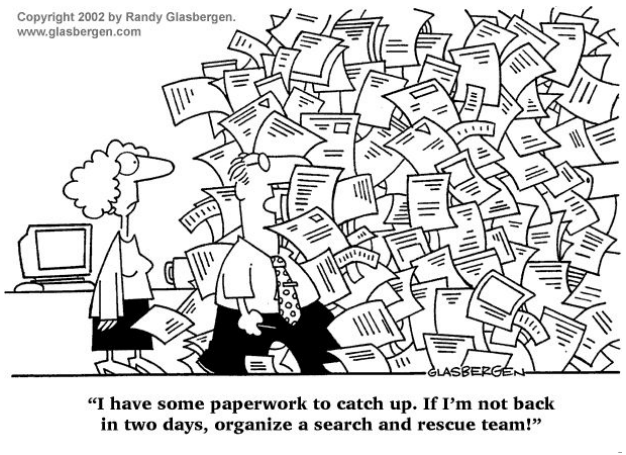


Filing


BIZARRO.COM
©2019 BIZARRO STUDIOS
Facebook.com/BizarroComics
Dist. by King Features
Wayno & Piraro
11-2-19
NOV
BLUE BERRY
APPLE
PUMPKIN
FILLING CABINET

FILE HANDLING



"I have some paperwork to catch up. If I'm not back in two days, organize a search and rescue team!"

Files store data permanently in a storage device. With file handling, the output from a program can be stored in a file.

Using file handling we can store our data in Secondary memory (Hard disk).

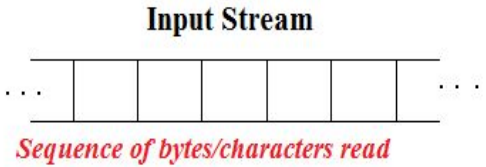
The transfer of input - data or output - data from one computer to another can be easily done by using files.

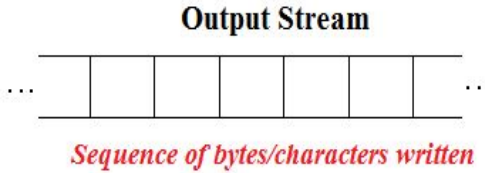
Filing

- It saves your data even if the program terminates.
- You can read a large amount of data using files.
- You can easily move your data from one computer to another without any changes.

Input Output Stream

Input Stream : It is flow of data bytes from a device (e.g Keyboard , disk drive) to main memory (when we read/take file's data into a program variable)





Output Stream : It is flow of data bytes from main memory (i.e program) to a device(when we store/write variable's data into a file)

File Operations

1. Creating a new file
2. Opening an existing file
3. Closing a file
4. Reading from and writing information to a file

Creating A File / Opening An Existing File

```
FILE *fptr;  // FILE is datatype

//fptr = fopen("fileopen","mode");
fptr = fopen("C:\\program.txt","w");

fclose(fptr);
```

Sr.No.	Mode & Description
1	**r** Opens an existing text file for reading purpose.
2	**w** Opens a text file for writing. If it does not exist, then a new file is created. Here your program will start writing content from the beginning of the file.
3	**a** Opens a text file for writing in appending mode. If it does not exist, then a new file is created. Here your program will start appending content in the existing file content.
4	**r+** Opens a text file for both reading and writing.
5	**w+** Opens a text file for both reading and writing. It first truncates the file to zero length if it exists, otherwise creates a file if it does not exist.
6	**a+** Opens a text file for both reading and writing. It creates the file if it does not exist. The reading will start from the beginning but writing can only be appended.

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

   fptr = fopen("program.txt","w");

   if(fptr == NULL)
    {
       printf("Error!");
    }
    fclose(fptr);

   return 0;
}
```

Writing To A File

```
fputc(char, file_pointer)

fputs(str, file_pointer)

fprintf(file_pointer, str, variable_lists)
```

"We have a VP of Records Management, but we don't know who it is because nobody can locate the file."

https://www.onlinegdb.com/online_c_compiler

```c
#include <stdio.h>
int main()
{
  int num;
  FILE *fptr;
  fptr = fopen("C:\\program.txt","w");
  if(fptr == NULL)
   {
      printf("Error!");
   }
  else
   {
      printf("Enter num: ");
      scanf("%d",&num);
      fprintf(fptr,"You entered %d\n Happy Coding",num);}
   fclose(fptr);
  return 0;   }
```

Reading From A File

```
fgetc(file_pointer)

fgets(buffer, count, file_pointer)

fscanf(file_pointer, str, variable_lists)
```

```c
#include <stdio.h>
int main()
{
  int num; FILE *fptr; char c;
  fptr = fopen("program.txt","r");
  if(fptr == NULL)
   {
      printf("Error!");
   }
  else
   {
      while ((c = fgetc(fptr)) != EOF)
          printf("%c", c);
   }
   fclose(fptr);
  return 0;
}
```

```c
#include <stdio.h>
int main()
{
  int num; FILE *fptr; char buffer[50];
   fptr = fopen("program.txt","r");
   if(fptr == NULL)
    {
       printf("Error!");
    }
   else
    {
       fgets(buffer,50,fptr); // It reads a single line
       printf("%s", buffer);
    }
    fclose(fptr);
   return 0;
}
```

```c
#include <stdio.h>
int main()
{
  int num; FILE *fptr; char buffer[50];
   fptr = fopen("program.txt","r");
   if(fptr == NULL)
    {
       printf("Error!");
    }
   else
    {
       while (fgets(buffer, sizeof(buffer), fptr) != NULL)
      {   printf("%s", buffer); // Print the current line }
    }
    fclose(fptr);
   return 0;
}
```

```c
#include <stdio.h>
int main()
{
  int num; FILE *fptr; char c[100], d[100];
  fptr = fopen("program.txt","r");
  if(fptr == NULL)
   {
      printf("Error!");
   }
  else
   {
      fscanf(fptr,"%s %s %d",&c, &d, &num);
      printf("%s %s %d", c,d,num);
   }
   fclose(fptr);
  return 0;
}
```

FSEEK

whence defines the point with respect to where the file pointer needs to be moved. It is specified by one of the following constants:

- SEEK_END: End of the file.
- SEEK_SET: Beginning of the file.
- SEEK_CUR: Current position of the file pointer.

```
fseek(fptr, 0, SEEK_END);
fseek(fptr, 10, SEEK_SET);
```

```c
#include <stdio.h>
int main()
{
  int num; FILE *fptr; char c[100], d[100];
  fptr = fopen("program.txt","r+");
  if(fptr == NULL)
      printf("Error!");
  else
   {
      fscanf(fptr,"%s %s %d",&c, &d, &num);
      printf("%s %s %d", c,d,num);
      fseek(fptr, 0, SEEK END);
      fprintf(fptr,"\nI am the new last line :)");
   }
   fclose(fptr);
  return 0;
}
```

Class task

Write a program to read a file and display contents with its line numbers.

Code

```
#include <stdio.h>
int main() {
    FILE *fptr;
    char buffer[256];
    int lineNumber = 1;
    fptr = fopen("program.txt", "r");
    if (fptr == NULL) {
        printf("Error: Could not open file!\n");
        return 1; // Exit with an error code
    }
    while (fgets(buffer, sizeof(buffer), fptr) != NULL) {
        printf("%d: %s", lineNumber, buffer);
        lineNumber++; // Increment line number
    }
    fclose(fptr);
    return 0;
}
```

Class task

Make a factorial program which takes input from a file input.txt and saves the output in another file result.txt.

Home Assignment

Make a program which reads all the text from a file but saves only count of vowels to another file.

Working With Structures

TXT files

Code

```
#include <stdio.h>
#include <string.h>
struct Student {
    int id;     char name[50];   float GPA; };

int main() {
    FILE *file;
    file = fopen("students.txt", "w");
    if (file == NULL) {  printf("Failed to open file for writing!\n");
                         return 1;  }
    struct Student s1 = {1, "John", 3.8};
    struct Student s2 = {2, "Ali", 3.5};
    fprintf(file, "%d %s %.1f\n", s1.id, s1.name, s1.GPA);
    fprintf(file, "%d %s %.1f\n", s2.id, s2.name, s2.GPA);
    fclose(file);
```

Code

```
    file = fopen("students.txt", "r+");
    if (file == NULL) {
        printf("Failed to open file for updating!\n");
        return 1;
    }
    struct Student s_read;
    int line = 0;
    fscanf(file, "%d %s %f", &s_read.id, s_read.name, &s_read.GPA);
    printf("Student: ID-%d, Name-%s, GPA-%.1f\n", s_read.id, s_read.name, s_read.GPA);
    line++;
    fscanf(file, "%d %s %f", &s_read.id, s_read.name, &s_read.GPA);
    printf("Student: ID-%d, Name-%s, GPA-%.1f\n", s_read.id, s_read.name, s_read.GPA);
    line++;
    fclose(file);   return 0;
}
```

Working With Structures

.DAT files

Binary Files

```
struct Student s1 = {1, "John", 3.8};

fwrite(&s1, sizeof(struct Student), 1, file);
```

Class task

Write a function read_data which takes input of a student struct and save the record in a file. File ptr is an argument of the function.

Code

```
void read_data(const char *filename) {
    FILE *file = fopen(filename, "ab"); // Open file in append-binary mode
    if (file == NULL) {
        printf("Error: Could not open file %s for writing.\n", filename);
        return;
    }

    struct Student s;
    printf("Enter Student ID: ");    scanf("%d", &s.id);
    printf("Enter Student Name: ");  scanf("%s", s.name);
    printf("Enter Student GPA: ");   scanf("%f", &s.GPA);
    fwrite(&s, sizeof(struct Student), 1, file);
    printf("Student record saved successfully.\n");
    fclose(file); // Close the file}
```

Class task

Write a function display_data which takes a file pointer as an input and displays all student records on console.

Code

```
// Function to display all student records
void display_data(const char *filename) {
    FILE *file = fopen(filename, "rb");
    if (file == NULL) {
        printf("Error: Could not open file %s for reading.\n", filename);
        return;
    }
    struct Student s;
    printf("Student Records:\n");
    printf("ID\tName\t\tGPA\n");
    while (fread(&s, sizeof(struct Student), 1, file)) {
        printf("%d\t%-10s\t%.2f\n", s.id, s.name, s.GPA);
    }
    fclose(file);}
```

Home Task

Update Data

Delete Data

Very important to do it by yourself.

References

https://www.guru99.com/c-file-input-output.html

https://www.educative.io/edpresso/what-is-fseek-in-c