File Functions

int fscanf(FILE *stream, const char* format, ...)	reads formatted input from a stream.
int fprintf(FILE *stream, const char* format, ...)	sends formatted output to a stream.
FILE *fopen(const char* filename, const char *mode)	opens the filename pointed to, by filename using the given mode. "r" Opens a file for reading. The file must exist. "w" Creates an empty file for writing. If a file with the same name already exists, its content is erased and the file is considered as a new empty file. "a" Appends to a file. Writing operations, append data at the end of the file. The file is created if it does not exist.
int fseek(FILE *f, long int offset, int origin)	go forward offset times without reading the files origins: SEEK_SET - from start of file SEEK_END - end of file SEEK_CUR - move from current location
int ftell(FILE *f)	returns the distance from cursor to start of file
void rewind(FILE *f)	go back to start of file
int ferror(FILE *F)	returns 0 if no errors occured
fclose(FILE* F)	closes the file
size_t fread(void *ptr, size_t size, size_t nmemb, FILE* stream)	reads data from the given stream into the array pointed to, by ptr.
int fgetc(FILE *stream)	Gets the next character (an unsigned char) from the specified stream and advances the position indicator for the stream.
char *fgets(char* str, int n, FILE *stream)	read line to str. stop when newline or eof is read
int fputc(int char, FILE *stream)	Writes a character specified by the argument char to the specified stream and advances the position indicator for the stream.
int fputs(const char *str, FILE* stream)	Writes a string to the specified stream up to but not including the null character.

Reading and writing binary files

Reading file into struct (cont)

```
>        people = realloc(people, sizeof(pe-
rson)*(++age));size);
        strcpy((people+size-1)->name, name);
        (people+size-1)->age = age;
    }
    fclose(f);
}
```

assume people.txt is formatted like this:
name1 age
name2 age

Writing to file

```
#include <stdio.h>
#define SIZE 20
struct Person{
char name[S IZE];
long id;
float age;
} typedef person_t;
void main(){
person_t p1={"mo mo", 1111,
23.5}, p2 = {"go go", 2222,
24.8}, p3, p4;
FILE* f = fopen( " per son s.b -
in", " wb");
fwrite (&p1, sizeof (pe rso -
n_t), 1, f);
fwrite (&p2, sizeof (pe rso -
n_t), 1, f);
fclose(f);
f = fopen( " per son s.b in",
" rb");
fread( &p3, sizeof (pe rso n_t),
1, f);
fread( &p4, sizeof (pe rso n_t),
1, f);
fclose(f);
printf ("p3: name: %s\t id:
%ld\t age: %.2f\n ", p3.name,
p3.id, p3.age);
printf ("p4: name: %s\t id:
%ld\t age: %.2f\n ", p4.name,
p4.id, p4.age);
}
```

```
#include <stdio.h>
void main()
{
FILE* f = fopen( " myF ile.tx -
t", " w");
int res;
if (f == NULL){
printf ("Failed opening the
file. Exitin g! \n");
return;
}
fputs( " Hello World! \n", f);
fclose(f);
f = fopen( " myF ile.tx t", " -
a");
fputs( "And Good Mornin g! \n",
f);
fclose(f);
}
```

Reading file into struct

```
typedef struct person {
       char name[20];
       int age;
} person;
int main() {
        person *people = NULL;
       char name[20];
       int size, age = 0;
       FILE *f = fopen( " peo -
ple.tx t", " r");
        while (!feof (f)){
                sca nf( "%s %d",
name, &age);
```