Structures || File Handling

Course: Introduction to Programming and Data Structures

Laltu Sardar

Institute for Advancing Intelligence (IAI),
TCG Centres for Research and Education in Science and Technology (TCG Crest)

tcg crest
Inventing Harmonious Future

August 27, 2024

Structures in C

tcg crest
Inventing Harmonious Future

Introduction to Structures

- A ‘struct‘ in C is a user-defined data type that allows grouping of variables of different types.
- Structures are used to represent a record and organize complex data.
- Defined using the ‘struct‘ keyword.

tcg crest
Inventing Harmonious Future

Defining a Structure

- A structure is defined as follows:

```
1  struct Person {
2    char name[50];
3    int age;
4    float salary;
5  };
```

- Another example:

```
1  struct Point {
2    int x;
3    int y;
4  };
```

tcg crest
Inventing Harmonious Future

Accessing Structure Members

- Structure members are accessed using the dot operator '.'.
- Example:

```
1    struct Person p1;
2    p1.age = 30;
3
4    struct Point p2;
5    p2.x = 10;
6    p2.y = 20;
```

tcg crest
Inventing Harmonious Future

Initializing Structures

- Structures can be initialized at the time of declaration.

```
1    struct Person p1 = {"John", 30, 50000.0};
```

tcg crest
Inventing Harmonious Future

Structure Declaration with Typedef

- Using ‘typedef‘ to simplify structure usage:

```
1    typedef struct {
2      int x;
3      int y;
4    } Point;
5
6    Point p1, p2;
```

tcg crest
Inventing Harmonious Future

Structure as Function Argument

- Structures can be passed to functions by value or by reference.
- Passing by reference is more efficient.

tcg crest
Inventing Harmonious Future

Example: Function with Structure Argument

```
1    void printPerson(struct Person p) {
2      printf("%s is %d years old and earns %.2f",
3      p.name, p.age, p.salary);
4    }
```

tcg crest
Inventing Harmonious Future

Structures and Pointers

- Pointers can be used to access structure members.
- The arrow operator (‘->’) is used to access members via a pointer.

tcg crest
Inventing Harmonious Future

Example: Pointer to Structure

```
1       struct Person *ptr;
2       ptr = &p1;
3       printf("%s", ptr->name);
```

Nested Structures

- Structures can be nested within other structures.
- Useful for representing complex data models.

tcg crest
Inventing Harmonious Future

Example: Nested Structures

```
1   struct Address {
2     char city[30];
3     int zip;
4   };
5
6   struct Person {
7     char name[50];
8     struct Address addr;
9   };
```

tcg crest
Inventing Harmonious Future

Self-Referential Structures

- Structures can have pointers to instances of the same structure.
- Commonly used in linked lists.

```
1  struct Node {
2    int data;
3    struct Node* next;
4  };
```

tcg crest
Inventing Harmonious Future

Bit Fields in Structures

- Structures can contain bit fields.
- Bit fields allow allocation of a specific number of bits for a variable.

```
1  struct Flags {
2    unsigned int flag1 : 1;
3    unsigned int flag2 : 1;
4    unsigned int flag3 : 2;
5  };
```

tcg crest
Inventing Harmonious Future

Example: Bit Fields in C Structure

```c
struct DeviceStatus { // Define a structure with bit fields
  unsigned int powerOn      : 1;
  unsigned int connected    : 1; //(0 or 1)
  unsigned int error        : 1; // (0 or 1)
  unsigned int mode         : 2; // (values 0 to 3)
  unsigned int reserved     : 3; // (values 0 to 7)
};

int main() {
  struct DeviceStatus status = {1, 0, 0, 2, 0};

  status.connected = 1; // Update the status
  status.error = 1;
  status.mode = 3;

  printf("\nUpdated Status:\n"); // Print the updated status
  printf("Power On: %d\n", status.powerOn);
  printf("Connected: %d\n", status.connected);
  printf("Mode: %d\n", status.mode);
  return 0;
}
```

Basics of File Handling in C

fscanf and fprintf

- fscanf and fprintf works almost same as scanf and printf

```c
// Program to learn basic file operation
#include<stdio.h>

float average(float a, float b){
    return ((a+b)/2.0);
}

int main(){
    float a, b, avg;

    FILE * inp_file_ptr, * out_file_ptr; //File type pointer must
        be declared

    inp_file_ptr = fopen("input_file.txt","r"); // Opening input
        file for reading
    fscanf(inp_file_ptr, "%f %f", &a, &b);  // taking input from
        file
    fclose(inp_file_ptr);   // closing the input file

    avg = average(a, b);    //Compauting avarage

```

File opening modes

• When you open a file, you need to specify the mode in which you want to open it. The following are the different file modes:

Mode	Meaning of Mode	During Inexistence of File
`r`	Reading.	If the file does not exist, `fopen()` returns `NULL`.
`w`	Writing.	If the file exists, its contents are overwritten. If the file does not exist, it will be created.
`a`	Append.	Data is added to the end of the file. If the file does not exist, it will be created.
`r+`	Reading and Writing.	If the file does not exist, `fopen()` returns `NULL`.
`w+`	Reading and Writing.	If the file exists, its contents are overwritten. If the file does not exist, it will be created.
`a+`	Reading and Appending.	If the file does not exist, it will be created.

Table: File opening modes in C

tcg crest
Inventing Harmonious Future

Reading from a file

Function	Description
`fscanf()`	Use formatted string and variable arguments list to take input from a file. `int fscanf(FILE *ptr, const char *format, ...)`
`fgets()`	Input the whole line from the file. `char *fgets(char *str, int n, FILE *stream)`
`fgetc()`	Reads a single character from the file. `int fgetc(FILE *pointer)`
`fread()`	Reads the specified bytes of data from a binary file. `size_t fread(void *ptr, size_t size, size_t nmemb, FILE *stream)`

Table: Some functions to Read from a file

tcg crest
Inventing Harmonious Future

Writing to a file

Function	Description
`fprintf()`	Similar to `printf()`, this function print output to the file. `int fprintf(FILE *fptr, const char *str, ...);`
`fputs()`	Prints the whole line in the file and a newline at the end. `int fputs(const char *str, FILE *stream)`
`fputc()`	Prints a single character into the file. `int fputc(int char, FILE *pointer)`
`fwrite()`	This function writes the specified amount of bytes to the binary file. `size_t fwrite(const void *ptr, size_t size, size_t nmemb, FILE *stream)`

Table: Some functions to Write from a file

tcg crest
Inventing Harmonious Future

Closing a file

1. The fclose() function is used to close the file
2. After successful file operations, you must always close a file **to remove it from the memory**.
3. Syntax of fclose()

```
fclose(file_pointer);
```

tcg crest
Inventing Harmonious Future

Example: Using `fseek` in C

The `fseek` function in C is used to move the file pointer to a specific location in a file. It is commonly used for random access in files.

Syntax:

- `int fseek(FILE *stream, long offset, int whence);`

Parameters:

- `stream` - Pointer to the file object.
- `offset` - Number of bytes to offset from `whence`.
- `whence` - Position from where `offset` is added.
  - `SEEK_SET` - Beginning of file.
  - `SEEK_CUR` - Current position of the file pointer.
  - `SEEK_END` - End of file.

tcg crest
Inventing Harmonious Future

Example Code Using fseek()

```c
int main() {
  FILE *fp;
  char c;

  // Open file in read mode
  fp = fopen("example.txt", "r");

  if (fp == NULL) {
    perror("Error opening file");
    return -1;
  }
  // Move the file pointer to the 10th byte from the beginning
  fseek(fp, 10, SEEK_SET);

  // Read and print the character at this position
  c = fgetc(fp);
  printf("Character at position 10: %c\n", c);


  fclose(fp); // Close the file
  return 0;
}
```