C Programming – Lecture VII

Instructor Özgür ZEYDAN
http://cevre.beun.edu.tr/

Two Dimensional Array Example

- Write a C program that calculates standart deviation of given numbers.
- Formula of standard deviation:

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2},$$

- You may use two dimensional array:
  std[N][3]

i	x_i	x_i-x_{ave}	$(x_i$-$x_{ave})^2$
0			
1			
2			
…			
…			
N-2			
N-1			

Characters in C

- `char` is a one-byte data type capable of holding a character.
- Characters in C consist of any printable or nonprintable character in the computer's character set including lowercase letters, uppercase letters, decimal digits, special characters and escape sequences.
  - 'a', 'b', 'c', …'z', '0', '1', … '9', '+', '-', '=', '!', '~', etc, '\n', '\0', etc.
- May be used in arithmetic expressions
  - Add, subtract, multiply, divide, etc.

- `char x='a';`

Strings in C

- A string is a character array terminated by '`\0`' .
- '`\0`' is a `NULL` character which indicates the end of the string.
- char a[]; OR char *a;
- char b[10] = "**`Hi there!`**";
- in memory :

H	i		t	h	e	r	e	!	\0

- char c[12] = "**`Hi there!`**";
- in memory :

H	i		t	h	e	r	e	!	\0	\0	\0

Placeholders in Strings

- You can **print string types with `printf`** by using different placeholders:
  - **char** (single character values) uses **`%c`**
  - **character strings** (arrays of characters) use **`%s`**

Example – 1

```
#include<stdio.h>
#include<conio.h>
void main()
{
char b[20]="C Programming";
int i;
printf("%s\n\n",b);
for(i=0;i<=13;i++)
        printf("%c\n",b[i]);
getch();
}
```

String Functions

- Requires `#include <string.h>`
- `strcpy (str1, str2);`
- Copies a string from a source address (str2) to a destination address (str1)

- char name[15];
- strcpy(name, "Ozgur hoca");
- printf("%s\n", name);

String Functions

- Requires `#include <string.h>`
- `strcat (str1, str2);`
- Takes two C-strings as input. Adds the contents of the second string (str2) to the end of the first string (str1).

- char str1[15] = "Hello ";
- char str2[30] = "Civil Engineers!";
- strcat(str1, str2);

- No automatic bounds checking: programmer must ensure that str1 has enough room for result.

String Functions

- Requires `#include <string.h>`
- `strlen(str1);`
- Returns length of a string.

- printf("%d",strlen("Ozgur hoca"));

- Screen output: 10

Example – 2

```
#include <string.h>
#include <stdio.h>
#include <conio.h>
void main()
{       char my_string[20];
char *blank = " ", *str1 = "Ozgur", *str2 = "Zeydan";
strcpy(my_string, str1);
strcat(my_string, blank);
strcat(my_string, str2);
printf("%s\n", my_string);
printf("%d",strlen(my_string));
getch();       }
```

File Commands in C

- C supports a number of functions that have the ability to perform basic file operations, which include:
  - Naming a file
  - Opening a file
  - Reading from a file
  - Writing data into a file
  - Closing a file

File Operation Functions in C

Function Name	Operation
fopen()	Creates a new file for use Opens a new existing file for use
fclose	Closes a file which has been opened for use
getc()	Reads a character from a file
putc()	Writes a character to a file
fprintf()	Writes a set of data values to a file
fscanf()	Reads a set of data values from a file
getw()	Reads a integer from a file
putw()	Writes an integer to the file
fseek()	Sets the position to a desired point in the file
ftell()	Gives the current position in the file
rewind()	Sets the position to the begining of the file

Opening Text Files

- Use **fopen** to open a file.
- It opens a file for a specified mode (the three most common are:
  - r: read
  - w: write
  - a: append
- It then returns a file pointer that you use to access the file.

- `FILE *myfile;`
- `myfile=fopen("output.txt","w");`

Opening Text Files

- `myfile=fopen("output.txt","w");`

- The **fopen** statement here opens a file named output.txt with the w mode.
- This is a destructive write mode, which means that
  - if out does not exist it is created,
  - but if it does exist it is destroyed and a new file is created in its place.

- The **fopen** command returns a pointer to the file, which is stored in the variable f. This variable is used to refer to the file.

Opening Text Files with File Name

```
char filename[10];
printf("Write filename: ");
scanf("%s",filename);
fp=fopen(filename,"w");
```

File Opening Modes

- r : (reading from the file) If the file exists, loads it into memory and sets up a pointer which points to the first character in it. If the file doesn't exit it returns NULL.
- w : (writing to the file) If the file exists, its contents are overwritten. If the file doesn't exit, a new file is created. Returns NULL, if unable to open file.
- a : (appending new contents at the end of file) If the file exists, loads it into memory and sets up a pointer which points to the first character in it. If the file doesn't exist, a new file is created. Returns NULL, if unable to open file.

File Opening Modes

- r+ : (reading existing contents, writing new contents, modifying existing contents of the file) If it exists, loads it into memory and sets up a pointer which points to the first character in it. If file doesn't exist it returns NULL.
- w+ : (writing new contents, reading them back and modifying existing contents of the file) If the file exists, its contents are destroyed. If the file doesn't exist a new file is created. Returns NULL, if unable to open file.
- a+ : (reading existing contents, appending new contents to end of file. Cannot modify existing contents) If the file exists, loads it in to memory and sets up a pointer which points to the first character in it. If the file doesn't exist, a new file is created. Returns NULL, if unable to open file.

Writing & Reading Data

- Use **`fprintf()`** function to write data to a text file.
- Use **`fscanf()`** function to read data from a text file.

```
int x,a;
```

```
fprintf(myfile,"%d",x);
```

```
fscanf(myfile2,"%d",&a);
```

Closing Text Files

- When we have finished reading from the file, we need to close it.
- This is done using the function **`fclose( )`** through the statement,
- **`fclose ( myfile ) ;`**

Example – 3

```
#include <stdio.h>  #include <conio.h>
#define MAX 10
main(){
FILE *myfile;
int x;
myfile=fopen("output.txt","w");
if (!myfile) return 1;
for(x=1; x<=MAX; x++)    fprintf(myfile,"%d\n",x);
fclose(myfile);
printf("Writing file complete. Press anykey to exit.");
getch();        return 0;       }
```

"output.txt" file

Example – 4

- Modify your previous code so that:
  - Your program will open `output.txt` file as source.
  - Read data from each line.
  - Calculate square of each value and write these values into `output2.txt` file.
  - Then program closes two files and terminates itself.

Example – 4

```
#include <stdio.h>  #include <conio.h> #define MAX 10
int main()     {
FILE *f1, *f2;
int x,a;
f1=fopen("output.txt","r");    f2=fopen("output2.txt","w");
if (!f1 || !f2) return 1;
for(x=1; x<=MAX; x++)       {
     fscanf(f1,"%d",&a);          fprintf(f2,"%d\n",a*a);      }
fclose(f1);         fclose(f2);
printf("Writing file complete. Press anykey to exit.");
getch();            return 0;   }
```

"output2.txt" file