**Arrays and Pointers Assignment**

1. What does the code below refer to? Extend the code and demonstrate the use of ptr to access the contents of a 2D array.

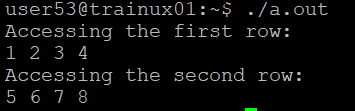
int (\*ptr)[4];

[Refer the sample code in “array\_ptr\_simple.c”]

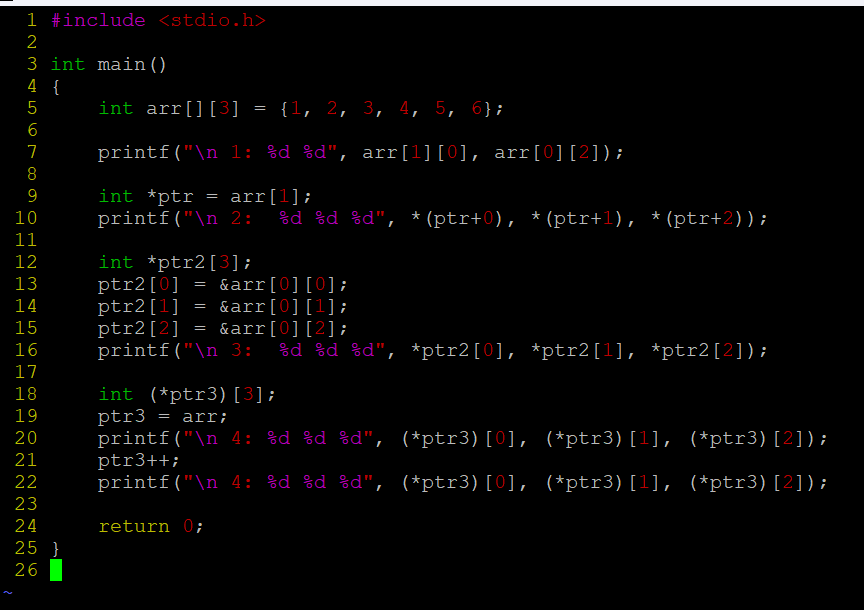
A computer screen shot of text

Description automatically generated

OUTPUT:



2. Refer the code in “array\_ptr\_simple\_char.c”. Implement the missing functionality in the code marked with TBD1, TBD2…..



A screen shot of a computer

Description automatically generated

3. Refer the code snippet below. Implement the function search\_insert() as mentioned in the code.

#define MAX 80

//search for the given char and if found, then create space for 1 character and insert ‘\_’ after the searched character. Let the remaining characters in the input be placed after ‘\_’.

int search\_insert(char name[], char search\_char);

int main()

{

char name[MAX]=”ABC”;

char \*ptr = name;

int ret = search\_insert(name, search\_char);

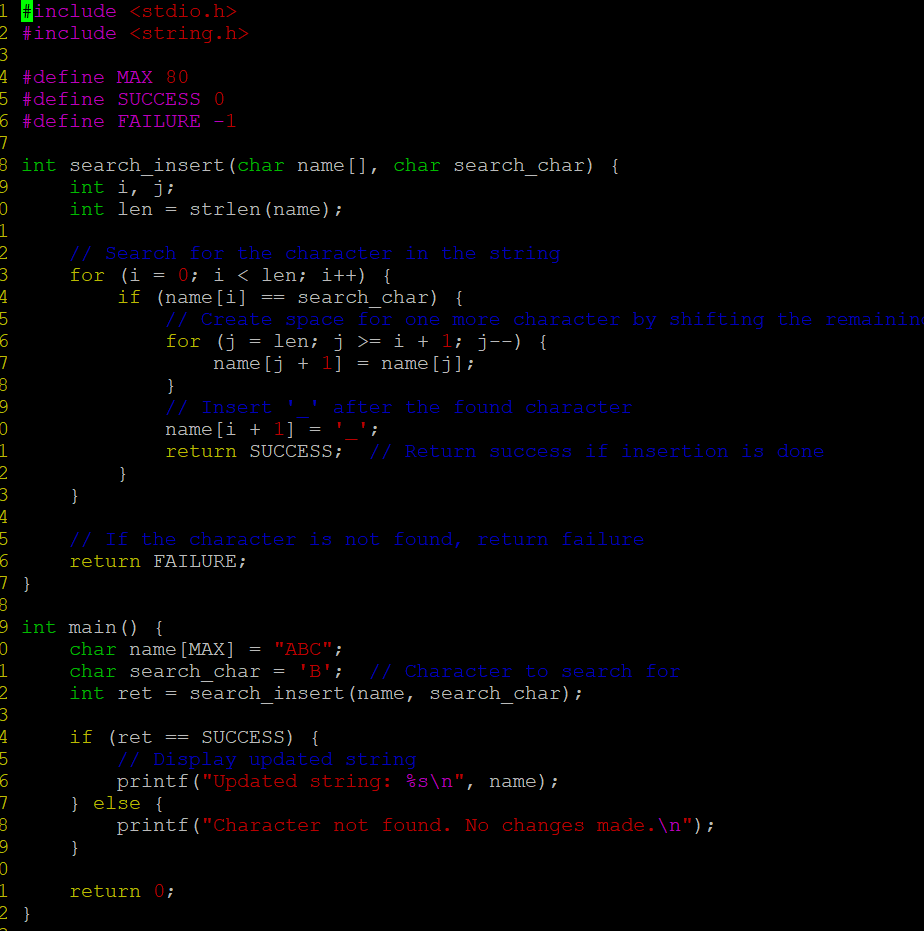
if (ret == SUCCESS)

{

//display updated string

}

}

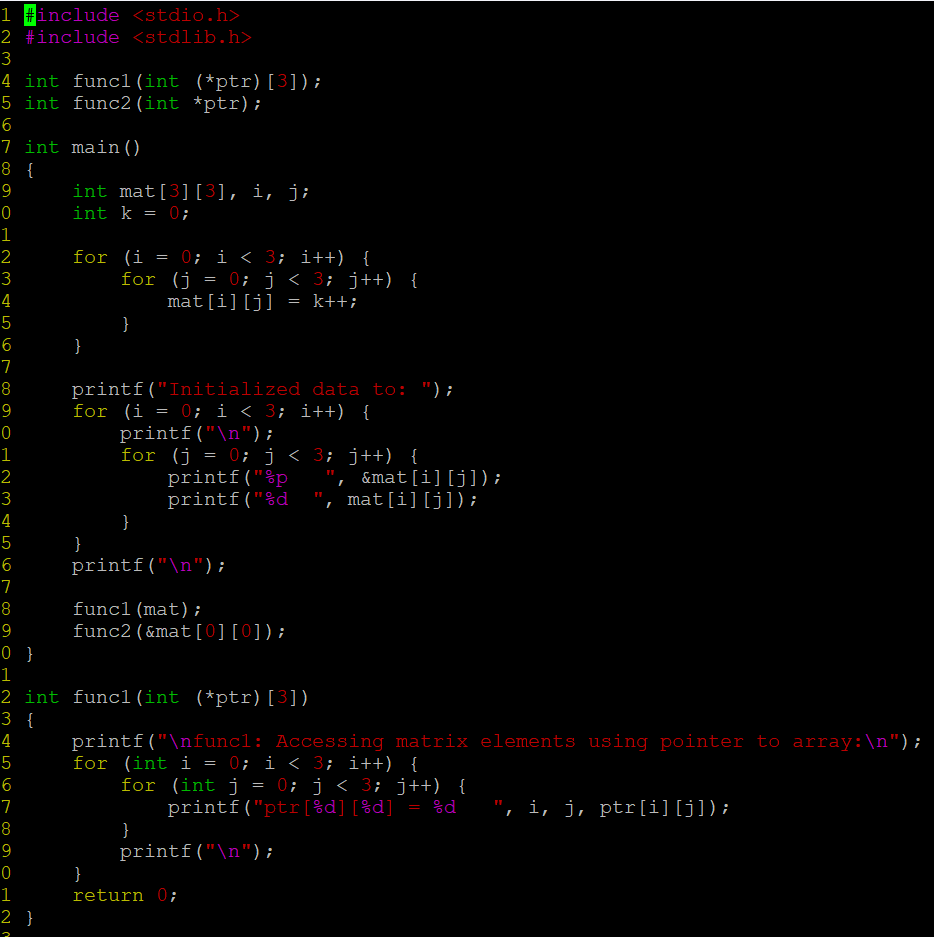




4. Refer the program “array\_ptr\_repr\_partial.c”. Implement the functions below which are yet to be implemented in code.

int func1(int (\*ptr)[3]); // pointer to array, second dimension is explicitly specified

int func2(int \*\*ptr); // double pointer, using an auxiliary array of pointers



A screenshot of a computer screen

Description automatically generated

OUTPUT:

A screenshot of a computer screen

Description automatically generated

5. Refer the program “array\_dbl\_pointers\_function\_partial.c”. Implement the missing functionality in the code marked with TBD1, TBD2…..



A screen shot of a computer program

Description automatically generated

A screenshot of a computer program

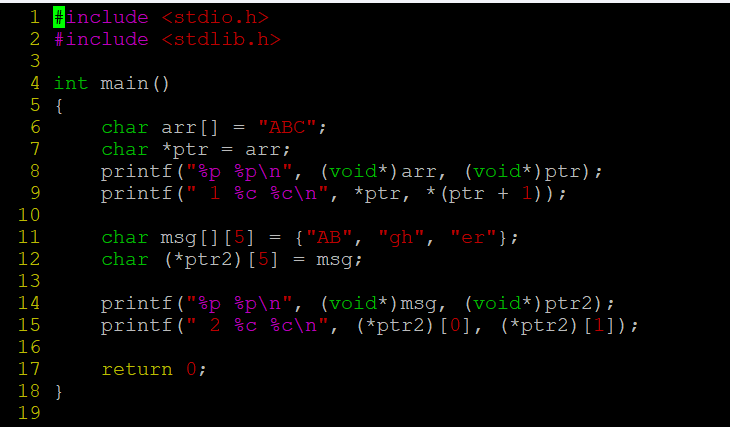
Description automatically generated

OUTPUT:

A screenshot of a computer screen

Description automatically generated

6. Refer the program "pointer\_example.c". Fix the warning issue.



OUTPUT:

A black screen with white text

Description automatically generated

7. Consider an array of strings as below.

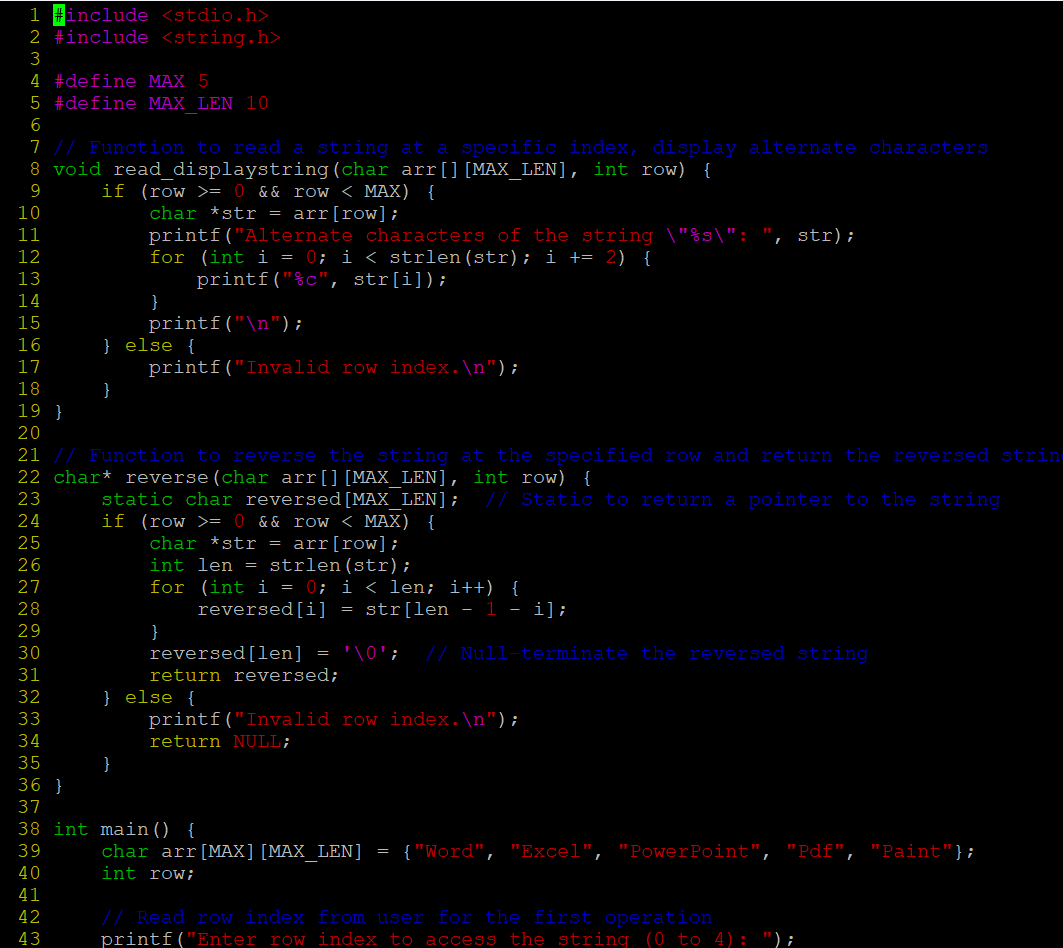
char arr[][10]={"Word", "Excel", "PowerPoint", "Pdf", "Paint"};

a. Implement a function read\_displaystring() to read a row index from the user, access the string, store in a char \* variable and using this, traverse every alternate character in the string and display in console.

void read\_displaystring(char \*arr[][10], int row);

b. Reverse the string read at the index in a) using a function of prototype as below. Caller to read the returned string and display the reversed string. [Ensure that the input source array is not corrupted and remaining elements are intact]

char \*reverse(char \*arr[][10], int row);



A screen shot of a computer code

Description automatically generated

OUTPUT:

A black screen with white text

Description automatically generated