

**SYNTAX**

1. Write a program using pointers to read in an array of integers and print its elements in reverse order.
2. We know that the roots of a quadratic equation of the form ax 2 + bx + c = 0
3. Write a function that receives a sorted array of integers and an integer value, and inserts the value in its correct place.
4. Write a function using pointers to add two matrices and to return the resultant matrix to the calling function.
5. Using pointers, write a function that receives a character string and a character as argument and deletes all occurrences of this character in the string. The function should return the corrected string with no holes.
6. Write a function day Aname that receives a number n and returns a pointer to a character string containing the name of the corresponding day. The day names should be kept in a static table of character strings local to the function.

7.Given an array of sorted list of integer numbers, write a function to search for a particular item, using the method of binary search. And also show how this function may be used in a program. Use pointers and pointer arithmetic.

8.Write a function (using a pointer parameter) that reverses the elements of a given array.

1. Write a function (using pointer parameters) that compares two integer arrays to see whether they are identical. The function returns 1 if they are identical, 0 otherwise.
2. Write a C program to insert a substring into another string by using function and pointers.
3. Write a C program that uses the pointer increment operations to demonstrate the scale factor.
4. Write a C program that displays the addresses and values pointed by an array of pointers.
5. Write a C program that uses pointers to demonstrate initialization of structure members.

1. Write a program using pointers to read in an array of integers and print its elements in reverse order.

Ans:

#include<stdio.h>

int main() {

int n;

printf("Enter the size of the array: ");

scanf("%d", &n);

int arr[n];

printf("Enter %d integers:\n", n);

for (int i = 0; i < n; i++) {

scanf("%d", &arr[i]);

}

printf("Array in reverse order:\n");

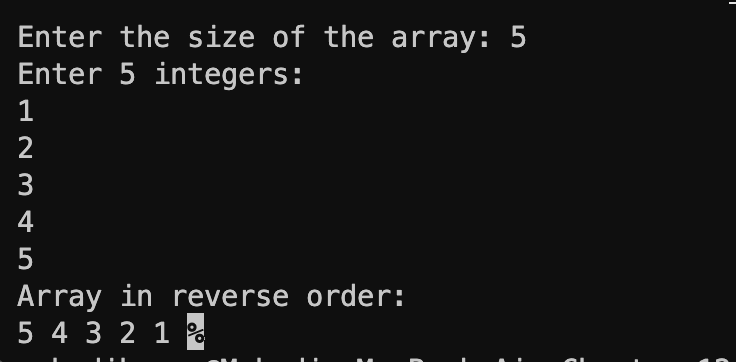
for (int i = n - 1; i >= 0; i--) {

printf("%d ", arr[i]);

}

return 0;

}



2. We know that the roots of a quadratic equation of the form ax 2 + bx + c = 0

Ans:

#include<stdio.h>

#include<math.h>

int main()

{

double a, b, c;

double discriminant, root1, root2;

printf("Enter coefficients a, b, and c: ");

scanf("%lf %lf %lf", &a, &b, &c);

discriminant = b \* b - 4 \* a \* c;

double \*pRoot1 = &root1;

double \*pRoot2 = &root2;

if (discriminant > 0) {

\*pRoot1 = (-b + sqrt(discriminant)) / (2 \* a);

\*pRoot2 = (-b - sqrt(discriminant)) / (2 \* a);

printf("Roots are real and different.\n");

printf("Root 1 = %.2lf\n", \*pRoot1);

printf("Root 2 = %.2lf\n", \*pRoot2);

} else if (discriminant == 0) {

\*pRoot1 = \*pRoot2 = -b / (2 \* a);

printf("Roots are real and same.\n");

printf("Root 1 = Root 2 = %.2lf\n", \*pRoot1);

} else {

double realPart = -b / (2 \* a);

double imaginaryPart = sqrt(-discriminant) / (2 \* a);

\*pRoot1 = realPart;

\*pRoot2 = imaginaryPart;

printf("Roots are complex and different.\n");

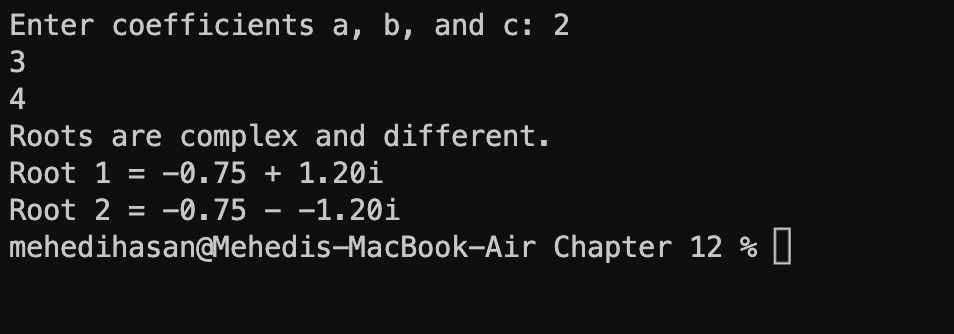
printf("Root 1 = %.2lf + %.2lfi\n", \*pRoot1, \*pRoot2);

printf("Root 2 = %.2lf - %.2lfi\n", \*pRoot1, -\*pRoot2);

}

return 0;

}



3. Write a function that receives a sorted array of integers and an integer value, and inserts the value in its correct place.

Ans: #include<stdio.h>

void insertIntoSortedArray(int arr[], int n, int value) {

int i = n - 1;

while (i >= 0 && arr[i] > value) {

arr[i + 1] = arr[i];

i--;

}

arr[i + 1] = value;

}

int main() {

int n;

printf("Enter the size of the sorted array: ");

scanf("%d", &n);

int sortedArray[n];

printf("Enter %d sorted integers:\n", n);

for (int i = 0; i < n; i++) {

scanf("%d", &sortedArray[i]);

}

int value;

printf("Enter the integer value to insert: ");

scanf("%d", &value);

insertIntoSortedArray(sortedArray, n, value);

printf("Array after insertion:\n");

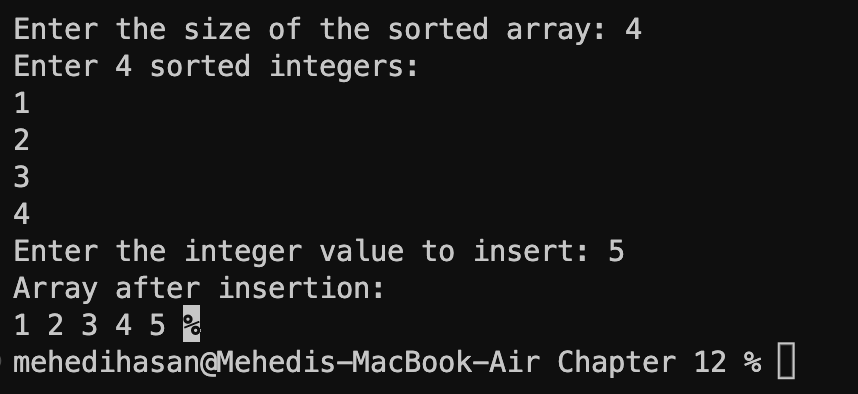
for (int i = 0; i <= n; i++) {

printf("%d ", sortedArray[i]);

}

return 0;

}



4. Write a function using pointers to add two matrices and to return the resultant matrix to the calling function.

Ans:

#include <stdio.h>

void addMatrices(int \*mat1, int \*mat2, int \*result, int rows, int cols)

{

int totalElements = rows \* cols;

for (int i = 0; i < totalElements; i++)

{

result[i] = mat1[i] + mat2[i];

}

}

void printMatrix(int \*matrix, int rows, int cols)

{

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < cols; j++)

{

printf("%d ", matrix[i \* cols + j]);

}

printf("\n");

}

}

int main()

{

int rows, cols;

printf("Enter the number of rows and columns: ");

scanf("%d %d", &rows, &cols);

int mat1[rows][cols]; int mat2[rows][cols];

int result[rows][cols];

printf("Enter elements of the first matrix:\n");

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < cols; j++)

{

scanf("%d", &mat1[i][j]);

}

}

printf("Enter elements of the second matrix:\n");

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < cols; j++)

{

scanf("%d", &mat2[i][j]);

}

}

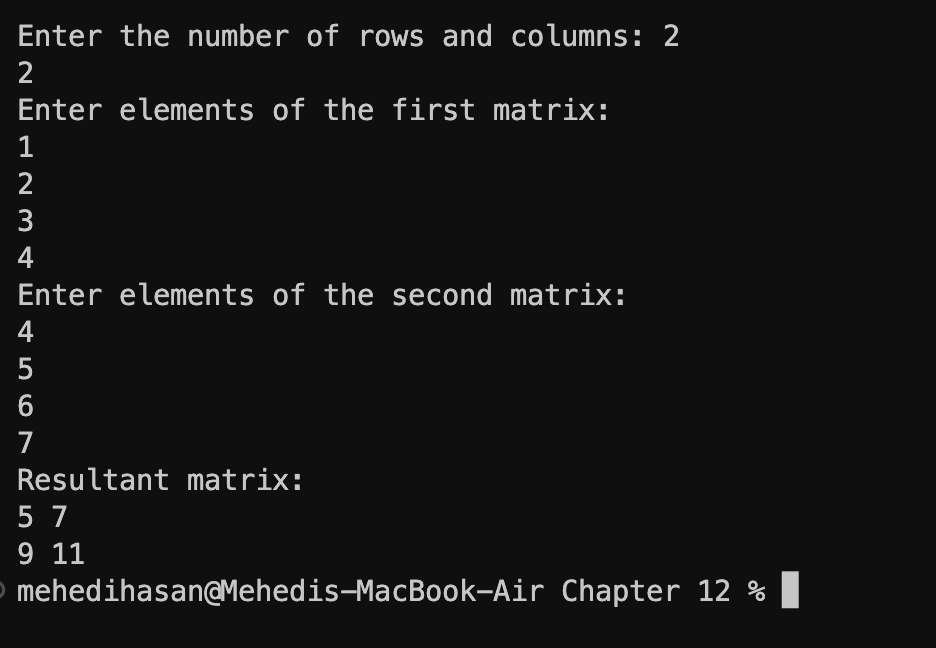
addMatrices((int \*)mat1, (int \*)mat2, (int \*)result, rows, cols);

printf("Resultant matrix:\n");

printMatrix((int \*)result, rows, cols);

return 0;

}



5. Using pointers, write a function that receives a character string and a character as argument and deletes all occurrences of this character in the string. The function should return the corrected string with no holes.

Ans:

#include <stdio.h>

char \*deleteCharacter(char \*str, char ch)

{

char \*temp = str;

char \*p = str;

while (\*temp)

{

if (\*temp != ch)

{

\*p = \*temp;

p++;

}

temp++;

}

\*p = '\0';

return str;

}

int main()

{

char str[] = "This is a test string";

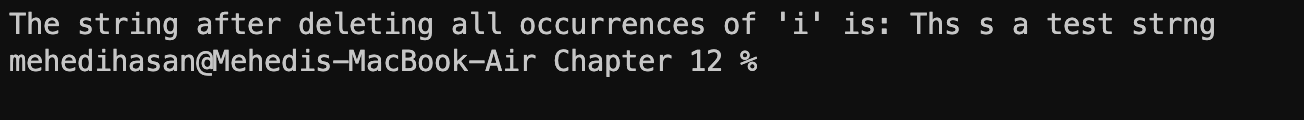
char ch = 'i';

char \*newStr = deleteCharacter(str, ch);

printf("The string after deleting all occurrences of '%c' is: %s\n", ch, newStr);

return 0;

}



6. Write a function day\_name that receives a number n and returns a pointer to a character string containing the name of the corresponding day. The day names should be kept in a static table of character strings local to the function.

Ans:

#include <stdio.h>

const char \*day\_name(int n) {

static const char \*days[] = {

"Invalid", "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"

};

if (n >= 1 && n <= 7) {

return days[n];

} else {

return days[0]; // Invalid day

}

}

int main() {

int n;

printf("Enter a number (1-7) to get the corresponding day: ");

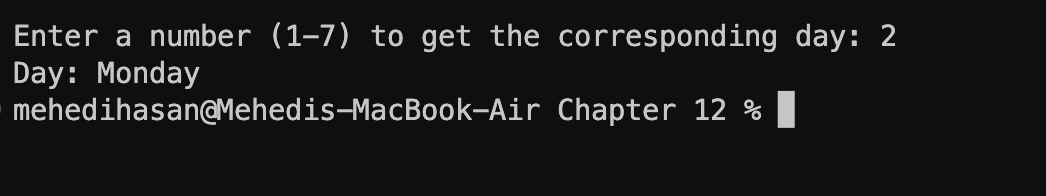
scanf("%d", &n);

const char \*day = day\_name(n);

printf("Day: %s\n", day);

return 0;

}



7. Given an array of sorted list of integer numbers, write a function to search for a particular item, using the method of binary search. And also show how this function may be used in a program.

Use pointers and pointer arithmetic.

Ans:

#include <stdio.h>

int binary search(const int \*arr, int size, int target) {

int left = 0;

int right = size - 1;

while (left <= right) {

int middle = left + (right - left) / 2;

if (arr[middle] == target) {

return middle;

} else if (arr[middle] < target) {

left = middle + 1;

} else {

right = middle - 1;

}

}

return -1;

}

int main() {

int sorted\_nums[] = {1, 4, 7, 9, 12, 15, 18, 21, 24, 27};

int size = sizeof(sorted\_nums) / sizeof(sorted\_nums[0]);

int target;

printf("Enter the number to search for: ");

scanf("%d", &target);

int index = binary\_search(sorted\_nums, size, target);

if (index != -1) {

printf("%d found at index %d\n", target, index);

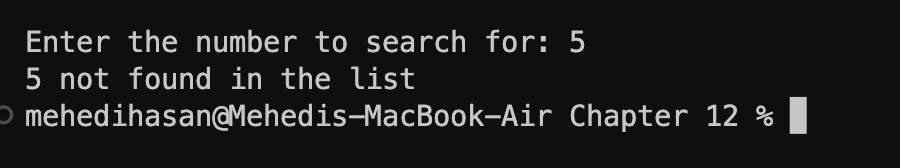
} else {

printf("%d not found in the list\n", target);

}

return 0;

}



1. Write a function (using a pointer parameter) that reverses the elements of a given array.

Ans: #include <stdio.h>

void reverseArray(int \*arr, int size) {

int start = 0;

int end = size - 1;

while (start < end) {

int temp = arr[start];

arr[start] = arr[end];

arr[end] = temp;

start++;

end--;

}

}

int main() {

int nums[] = {1, 2, 3, 4, 5, 6, 7, 8};

int size = sizeof(nums) / sizeof(nums[0]);

printf("Original array: ");

for (int i = 0; i < size; i++) {

printf("%d ", nums[i]);

}

printf("\n");

reverseArray(nums, size);

printf("Reversed array: ");

for (int i = 0; i < size; i++) {

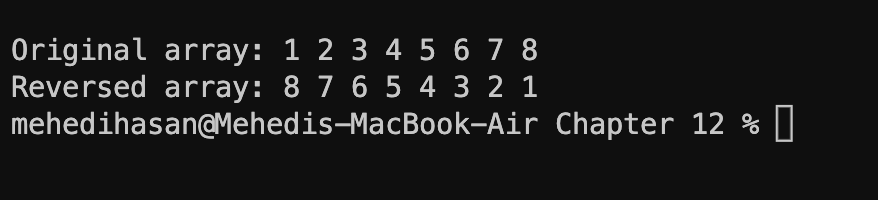
printf("%d ", nums[i]);

}

printf("\n");

return 0;

}



9. Write a function (using pointer parameters) that compares two integer arrays to see whether they are identical. The function returns 1 if they are identical, 0 otherwise.

Ans:

#include <stdio.h>

int compareArrays(int \*arr1, int \*arr2, int n)

{

for (int i = 0; i < n; i++)

{

if (arr1[i] != arr2[i])

{

return 0;

}

}

return 1;

}

int main()

{

int arr1[] = {1, 2, 3, 4, 5}; int

arr2[] = {1, 2, 3, 4, 5}; int n = sizeof(arr1) / sizeof(arr1[0]);

int result = compareArrays(arr1, arr2, n);

if (result == 1)

{

printf("The arrays are identical.\n");

} else

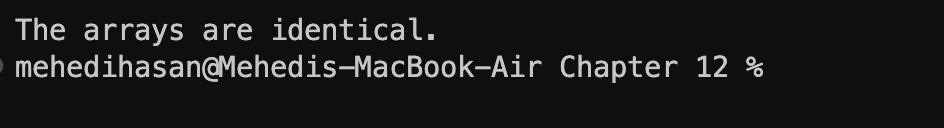
{

printf("The arrays are not identical.\n");

}

return 0;

}



10. Write a C program to insert a substring into another string by using function and pointers.

Ans:

#include <stdio.h>

#include <string.h>

void insertSubstring(char \*dest, const char \*src, int pos)

{

memmove(dest + pos + strlen(src), dest + pos, strlen(dest) - pos + 1);

strncpy(dest + pos, src, strlen(src));

}

int main() {

char mainStr[100] = "Hello, there!";

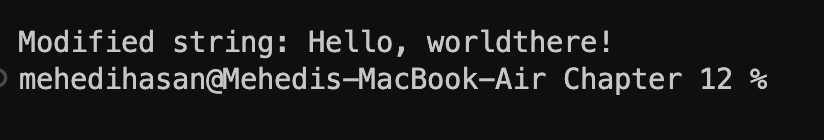
char subStr[] = "world";

insertSubstring(mainStr, subStr, 7);

printf("Modified string: %s\n", mainStr);

return 0;

}



11. Write a C program that uses the pointer increment operations to demonstrate the scale factor.

Ans:

#include <stdio.h>

int main() {

int numbers[] = {10, 20, 30, 40, 50};

int \*ptr = numbers;

printf("Original array:\n");

for (int i = 0; i < 5; i++) {

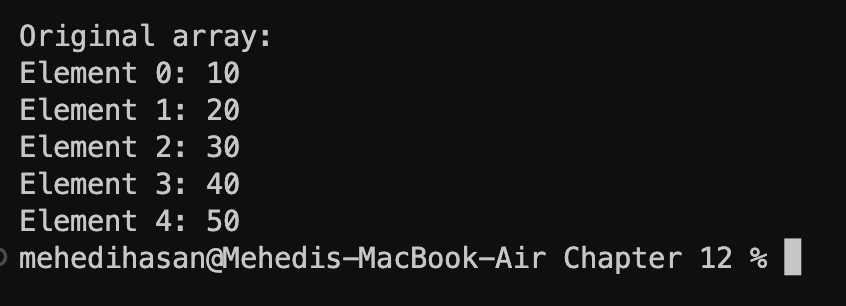
printf("Element %d: %d\n", i, \*ptr);

ptr++;

}

return 0;

}



12. Write a C program that displays the addresses and values pointed by an array of pointers. Ans:

#include <stdio.h>

int main() {

int num1 = 10, num2 = 20, num3 = 30;

int \*ptrArray[] = {&num1, &num2, &num3};

int numPointers = sizeof(ptrArray) / sizeof(ptrArray[0]);

printf("Array of Pointers:\n");

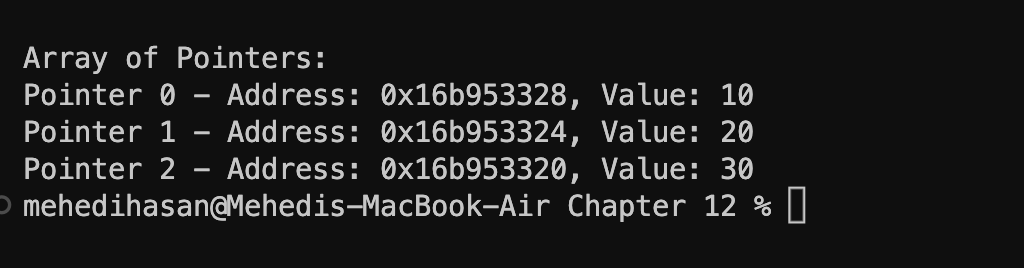
for (int i = 0; i < numPointers; i++) {

printf("Pointer %d - Address: %p, Value: %d\n", i, ptrArray[i], \*ptrArray[i]);

}

return 0;

}



13.Write a C program that uses pointers to demonstrate initialization of structure members

Ans:

#include <stdio.h>

#include <string.h>

struct Student {

int rollNumber;

char name[50];

int age;

};

int main() {

struct Student student1;

struct Student \*ptrStudent = &student1;

ptrStudent->rollNumber = 101;

strcpy(ptrStudent->name, "Mehedi");

ptrStudent->age = 20;

printf("Initialized Student:\n");

printf("Name: %s\n", ptrStudent->name);

printf("Roll Number: %d\n", ptrStudent->rollNumber);

printf("Age: %d\n", ptrStudent->age);

return 0;

}

