Practice – 6

1-D ARRAY

a. Write a C program to print the sum and free the memory where the array is stored

Program

```
#include<stdio.h>
#include  <stdio.h>
#include <stdlib.h>

int main() {
    int n, sum = 0;
    scanf("%d", &n);
    int *arr = (int *)malloc(n * sizeof(int));
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
        sum += arr[i];
    }
    printf("%d\n", sum);
    free(arr);
    return 0;
}
```

Input:

5 1 2 3 4 5

Output:

15

b.Write a C program of working with indices in array.

Program

```
#include <stdio.h>

int main() {
int n;
    scanf("%d", &n);
    int arr[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    for (int i = n - 1; i >= 0; i--) {
        printf("%d ", arr[i]);
    }
    return 0;
}
```

Input:

5 1 2 3 4 5

Output:

5 4 3 2 1

c.Write a C program to search an element in array (Linear Search)

Program

```
#include <stdio.h>
int main() {
    int n, key, found = 0;
    scanf("%d", &n);
    int arr[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);

    }

    scanf("%d", &key);
    for (int i = 0; i < n; i++) {
        if (arr[i] == key) {
            found = 1;
            printf("Element found at index %d\n", i);
            break;
        }
    }
    if (!found) {
        printf("Element not found\n");
    }
    return 0;
}
```

Input:

5 1 2 3 4 5 3

Output:

Element found at index 2

d.Write a C program to find min and max elements in array.

Program

```
#include <stdio.h>


int main() {
   int n;
   scanf("%d",  &n);
   int arr[n], min, max;
   scanf("%d", &arr[0]);
   min = max = arr[0];
   for (int i = 1; i < n; i++) {
      scanf("%d", &arr[i]);
      if (arr[i] < min) min = arr[i];
      if (arr[i] > max) max = arr[i];
   }
   printf("Min: %d\nMax: %d\n", min, max);
   return 0;
}
```

Input:

5 1 2 3 4 5

Output:

Min: 1 Max: 5

e.Write a C program to insert an element into array.

Program

```
#include <stdio.h>
int main() {
    int n, pos, elem;
    scanf("%d", &n);
    int arr[n + 1];
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    scanf("%d %d", &pos, &elem);
    for (int i = n; i > pos; i--) {
        arr[i] = arr[i - 1];
    }
    arr[pos] = elem;
    for (int i = 0; i <= n; i++) {
        printf("%d ", arr[i]);
    }
    return 0;
}
```

Input:

5 1 2 3 4 5 2 10

Output:

1 2 10 3 4 5

f.Write a C program to eliminate duplicate elements from array.

Program

```
#include <stdio.h>
int main() {
    int n, new_n = 0;
    scanf("%d", &n);
    int arr[n], result[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    for (int i = 0; i < n; i++) {
        int j;
        for (j = 0; j < new_n; j++) {
            if (arr[i] == result[j]) break;
        }
        if (j == new_n) {
            result[new_n++] = arr[i];
        }
    }
    for (int i = 0; i < new_n; i++) {
        printf("%d ", result[i]);
    }
    return 0;
}
```

Input:

6 1 2 2 3 3 4

Output:

1 2 3 4

g) Write a C program that Sorting of Elements in an Array using Bubble Sort.

Program

```
#include <stdio.h>
int main() {
   int n;
   scanf("%d", &n);
   int arr[n];
   for (int i = 0; i < n; i++) {
      scanf("%d", &arr[i]);
   }
   for (int i = 0; i < n-1; i++) {
      for (int j = 0; j < n-i-1; j++) {
         if (arr[j] > arr[j+1]) {
            int temp = arr[j];
            arr[j] = arr[j+1];
            arr[j+1] = temp;
         }
       }
    }
   for (int i = 0; i < n; i++) {
      printf("%d ", arr[i]);
   }
   return 0;
}
```

Input:

5 5 4 3 2 1

Output:

1 2 3 4 5

Practice – 7

2 – D ARRAY

a. Write a C program to print sum of two 2-D arraysProgram

```
#include<stdio.h>
int main()
{
  int r,c,a[10][10],i,j,b[10][10],t[10][10];
  printf("\nEnter the row limit:");
  scanf("%d",&r);
  printf("\nEnter the column limit:");
  scanf("%d",&c);
  printf("\nEnter the values:");
  for(i=0;i<r;i++)
  {
    for(j=0;j<c;j++)
    {
      scanf("%d",&a[i][j]);
    }
  }
  for(i=0;i<r;i++)
  {
    for(j=0;j<c;j++)
    {
      scanf("%d",&b[i][j]);
    }
  }
  for(i=0;i<r;i++)
  {
    for(j=0;j<c;j++)
    {
      t[i][j]=a[i][j]+b[i][j];
    }
  }
  printf("Addition Value:\n");
  for(i=0;i<r;i++)
  {
    for(j=0;j<c;j++)
    {
      printf("\t%d",t[i][j]);
    }
    printf("\n");
  }
  return 0;
}
```

Input:

Enter the row limit: 2

Enter the column limit: 2

Enter the values:2 4 5 6

1 2 3 4

Output:

Addition Value:

3 6

8 10

b.Multiplication of two 2-D arrays

```
#include<stdio.h>
int main()
{
    int a[3][3],b[3][3],c[3][3],i,j,r1,c1,r2,c2,k;
    printf("Enter order of matrix 1");
    scanf("%d%d",&r1,&c1);
    printf("Enter order of matrix 2");
    scanf("%d%d",&r2,&c2);
    if(c1==r2)
    {
        printf("Enter elements of first matrix-1");
        for(i=0;i<r1;i++)
        {
            for(j=0;j<c1;j++)
            {
                scanf("%d",&a[i][j]);
            }
        }
        printf("Enter elements of second matrix-2");
        for(i=0;i<r2;i++)
        {
            for(j=0;j<c2;j++)
            {
                scanf("%d",&b[i][j]);
            }
        }
```

```
            for(i=0;i<r1;i++)
            {
                  for(j=0;j<c2;j++)
                  {
                        c[i][j]=0;
                        for(k=0;k<c1;k++)
                        {
                              c[i][j]=c[i][j]+a[i][k]*b[k][j];
                        }
                  }
            }
            printf("After multiplication of result is : \n");
            for(i=0;i<r1;i++)
            {
                  for(j=0;j<c2;j++)
                  {
                        printf("%d \t",c[i][j]);
                  }
                  printf("\n");
            }
      }
      else
      {
            printf("Matrix multiplication is not possible");
      }
}
```

Input:

Enter order of matrix 1:3 3

Enter order of matrix 2:3 3

Enter elements of first matrix-1:1 2 3 4 5 6 7 8 9

Enter elements of second matrix-2:9 8 7 4 5 6 1 2 3

Output:

After multiplication of result is :

20 24 28

62 69 76

104 114 124

c.Write a C program to find transpose of a matrix.

Program

```
#include <stdio.h>
int main() {
    int m, n;
    scanf("%d %d", &m, &n);
    int arr[m][n];
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            scanf("%d", &arr[i][j]);
        }
    }
    int transpose[n][m];
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            transpose[j][i] = arr[i][j];
        }
    }
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            printf("%d ", transpose[i][j]);
        }
    printf("\n");
    }
    return 0;
}
```

Input:

2 3

1 2 3

4 5 6

Output:

1 4

2 5

2 6

d.Write a C program to find trace of a matrix.

Program

```
#include <stdio.h>

int main() {

    int n;

    scanf("%d", &n);

    int arr[n][n], trace = 0;
    for (int i = 0; i < n; i++) {

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

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

            if (i == j) trace += arr[i][j];

        }

    }
    printf("Trace: %d\n", trace);

    return 0;

}
```

Input:

3

1 2 3

4 5 6

7 8 9

Output:

Trace: 15

e.Write a C program to find lower triangular matrix.

Program

```
#include <stdio.h>

int main() {
    int n;
    scanf("%d", &n);
    int arr[n][n];
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            scanf("%d", &arr[i][j]);
        }
    }
    printf("Lower Triangular Matrix:\n");
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            if (i >= j) {
                printf("%d ", arr[i][j]);
            } else {
                printf("0 ");
            }
        }
        printf("\n");
    }
    return 0;
}
```

Input:

3

1 2 3

4 5 6

7 8 9

Output:

1 0 0

4 5 0

7 8 9

Practice – 8

STRINGS

a. Write a C program to print each word of the sentence in a new line.

Program

```
#include <stdio.h>
#include <string.h>
int main() {
    char str[100];
    gets(str);
    char *token = strtok(str, " ");
    while (token != NULL) {
        printf("%s\n", token);
        token = strtok(NULL, " ");
    }
    return 0;
}
```

Input:

Hello World This is C

Output:

Hello

World

This

Is

C

b. Write a C program to count number of alphabets (lowercase, uppercase, constants, vowels) and digits Lowercase to Uppercase, Uppercase to Lowercase, Toggle case, Sentential case.

Program

```
#include  <stdio.h>

#include <ctype.h>

#include <string.h>


void count_characters(char str[]) {

   int lowercase = 0, uppercase = 0, consonants = 0, vowels = 0, digits = 0;
   for (int i = 0; str[i] != '\0'; i++) {

      char ch = str[i];

      if (isdigit(ch)) {

         digits++;

      } else if (isalpha(ch)) {

         if (islower(ch)) lowercase++;

         if (isupper(ch)) uppercase++;


         ch = tolower(ch); // To check for vowels or consonants

         if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {

            vowels++;

         } else {

            consonants++;

         }

      }

   }
   printf("Character Counts:\n");

   printf("Lowercase: %d\n", lowercase);

   printf("Uppercase: %d\n", uppercase);

   printf("Vowels: %d\n", vowels);

   printf("Consonants: %d\n", consonants);

   printf("Digits: %d\n", digits);
```

```
    printf("\n");
}
void toggle_cases(char str[]) {
    printf("Case Transformations:\n");
// Lowercase to Uppercase
    printf("Lowercase to Uppercase:
    ");for (int i = 0; str[i] != '\0'; i++)
    {
        if (islower(str[i])) {
            printf("%c", toupper(str[i]));
        } else {
            printf("%c", str[i]);
        }
    }
    printf("\n");
    // Uppercase to Lowercase
    printf("Uppercase to Lowercase: ");
    for (int i = 0; str[i] != '\0'; i++) {
        if (isupper(str[i])) {
            printf("%c", tolower(str[i]));
        } else {
            printf("%c", str[i]);
        }
    }
    printf("\n");
    // Toggle case
    printf("Toggle Case: ");
```

```
    for (int i = 0; str[i] != '\0'; i++) {
        if (islower(str[i])) {
            printf("%c", toupper(str[i]));
        } else if (isupper(str[i])) {
            printf("%c", tolower(str[i]));
        } else {
            printf("%c", str[i]);
        }
    }
    printf("\n\n");
}
void sentential_case(char str[]) {
    int is_new_sentence = 1;

    printf("Sentential Case: ");
    for (int i = 0; str[i] != '\0'; i++) {
        if (is_new_sentence && isalpha(str[i])) {
            printf("%c", toupper(str[i]));
            is_new_sentence = 0;
        } else {
            printf("%c", tolower(str[i]));
        }
        if (str[i] == '.' || str[i] == '!' || str[i] == '?') {
            is_new_sentence = 1;
        }
    }
    printf("\n");
}
```

```
int main() {
    char str[100];

    printf("Enter a string: ");
    gets(str);

    // Count characters
    count_characters(str);

    // Perform case transformations
    toggle_cases(str);

    // Convert to Sentential Case
    sentential_case(str);

    return 0;

}
```

Input Example:

hello world. this is c! let's code?

Output Example:

Character Counts:

Lowercase: 28

Uppercase:0

Vowels: 8

Consonants: 20

Digits: 0

Case Transformations:

Lowercase to Uppercase: HELLO WORLD. THIS IS C! LET*'S CODE?*

Uppercase to Lowercase: hello world. this is c! let*'s code?*

Toggle Case: HELLO WORLD. THIS IS C! LET*'S CODE?*

Sentential Case: Hello world. This is c! Let*'s code?*

c. Write a C program to find the frequency of each digit in the given string.

Program

```
#include<stdio.h>
int main()
{
   char str[1001];
   int digit_count[10] = {0};
   scanf("%s", str);
   // Count occurrences of each digit
   for (int i = 0; str[i]; i++)
      if (str[i] >= '0' && str[i] <= '9')
         digit_count[str[i] - '0']++;

         // Print the results in a single line with spaces
   for (int i = 0; i < 10; i++)

      printf("%d%c", digit_count[i], i < 9 ? ' ' : '\n');
   return 0;
}
```

Input:

A11472o5tc

Output:

0 2 1 0 1 1 1 1 0 0

d.Write a C program to find String Length, Concatenate Two Strings, and Reverse a String using built-in and without built-in string functions.

Program

```
#include <stdio.h>
// Function to calculate string length without built-in functions
int string_length(char str[]) {
    int length = 0;
    while (str[length] != '\0') {
        length++;
    }
    return length;
}
// Function to concatenate two strings without built-in functions
void concatenate(char str1[], char str2[], char result[]) {
    int i = 0, j = 0;
    // Copy first string
    while (str1[i] != '\0') {
        result[i] = str1[i];
        i++;
    }
    // Append second string
    while (str2[j] != '\0') {
        result[i] = str2[j];
        i++;
        j++;
    }
```

```
    result[i] = '\0'; // Null-terminate the concatenated string
}

// Function to reverse a string without built-in functions
void reverse_string(char str[], char reversed[]) {
    int length = string_length(str);
    for (int i = 0; i < length; i++) {
        reversed[i] = str[length - i - 1];
    }
    reversed[length] = '\0'; // Null-terminate the reversed string
}

int main() {
    char str1[100], str2[100], result[200], reversed[100];

    // Find String Length
    printf("Enter a string to find its length: ");
    gets(str1);
    printf("Length using built-in function: %lu\n", strlen(str1));
    printf("Length without built-in function: %d\n\n", string_length(str1));

    // Concatenate Two Strings
    printf("Enter the first string for concatenation: ");
    gets(str1);
    printf("Enter the second string for concatenation: ");
    gets(str2);
    printf("Concatenated string using built-in function: %s\n", strcat(strcpy(result, str1), str2));
    concatenate(str1, str2, result);
    printf("Concatenated string without built-in function: %s\n\n", result);
```

```
    // Reverse a String
    printf("Enter a string to reverse: ");
    gets(str1);
    printf("Reversed string using built-in function: ");for (int
    i = strlen(str1) - 1; i >= 0; i--) {
        putchar(str1[i]);
    }
    printf("\n"); reverse_string(str1,
    reversed);
    printf("Reversed string without built-in function: %s\n", reversed);

    return 0;
}
```

Input:

Enter a string to find its length: Hello

Enter the first string for concatenation: Hello Enter the

second string for concatenation: WorldEnter a string to

reverse: Programming

Output:

Length using built-in function: 5 Length

without built-in function: 5

Concatenated string using built-in function: HelloWorld

Concatenated string without built-in function: HelloWorldReversed

string using built-in function: gnimmargorP Reversed string without

built-in function: gnimmargorP

Practice– 9

Functions and Recursion

(a) Functions in C

Objective: Learn simple usage of functions.
https://www.hackerrank.com/challenges/functions-in-c/problem?isFullScreen=true

```
#include <stdio.h>
int max_of_four(int a, int b, int c, int d)
{
    if (a > b && a > c && a > d) return a;
    else if (b > a && b > c && b > d) return b;
    else if (c > a && c > b && c > d) return c;
    return d;
}
 int main()  {
    int a, b, c, d;
    scanf("%d %d %d %d", &a, &b, &c, &d);
    int ans = max_of_four(a, b, c, d);
    printf("%d", ans);
    return 0;
}
```

Output:

```
 3   4 5 6
6
```

b.Fibonacci Numbers using recursion

```
#include <stdio.h>
int fibonacci(int n) {
        if (n <= 1)
        {
        return n;
    } else {
        return fibonacci(n - 1) + fibonacci(n - 2);
    }
}
int main()
{
    int n;
    printf("Enter the number of terms in the Fibonacci sequence: ");
    scanf("%d", &n);
    printf("Fibonacci Sequence:\n");
    for (int i = 0; i < n; i++) {
        printf("%d ", fibonacci(i));
    }
    return 0;
}
```

Output:

```
Enter the number of terms in the Fibonacci sequence: 13
Fibonacci Sequence:
    0    1 1 2 3 5 8 13 21 34 55 89 14
```

c.Factorial using recursion

```
#include<stdio.h>
int fact(int);int main()
{
     int x,n;
     printf(" Enter the Number to Find Factorial :");
     scanf("%d",&n);
     x=fact(n);
    printf(" Factorial of %d is %d",n,x);
return 0;}int fact(int n)
{
     if(n==0)
        return(1);
return(n*fact(n-1));
}
```

Output:

```
Enter the Number to Find Factorial :5
Factorial of 5 is 120
```

d.Find the digit sum using recursion

```
#include <stdio.h>
 int sum (int a);
 int main()
{
    int  num,  result;
    printf("Enter the number: ");
    scanf("%d", &num);
    result = sum(num);
    printf("Sum of digits in %d is %d\n", num, result);
    return 0;}
 int sum (int num)
{
    if (num != 0)
    {
       return (num % 10 + sum (num / 10));
    }
    else
    {
      return 0;
}
}
```

Output:

```
Enter the number: 2345
Sum of digits in 2345 is 14
```

e.LCM using recursive function

```
#include <stdio.h>
/* Function declaration */int lcm(int a, int b);
int main()
{
    int num1, num2, LCM;
    /* Input two numbers from user */
    printf("Enter any two numbers to find lcm: ");
    scanf("%d%d", &num1, &num2);
    /*
     * Ensures that first parameter of LCM function
     * is always less than second
     */
    if(num1 > num2)
       LCM = lcm(num2, num1);
    else
       LCM = lcm(num1, num2);

    printf("LCM of %d and %d = %d", num1, num2, LCM);
    return 0;}
   int lcm(int a, int b){ static int multiple = 0;
  if((multiple % a == 0) && (multiple % b == 0))
    {
       return multiple;
    }
    else
    {
       return lcm(a, b);
}
}
```

Output:

```
Enter any two numbers to find lcm: 12
30
LCM of 12 and 30 = 60
```

f.the Nth term using recursive function

https://www.hackerrank.com/challenges/recursion-in-c/problem?isFullScreen=true#include <stdio.h>

```
int find_nth_term(int n, int a, int b,
  int c) {int i;
  for (i = 1; i <
     n; i++) { int
     sum = a + b
     + c;a = b;
     b = c;
     c = sum;
  }
  return a;
}
int
    main
    () {
    int n,
    a, b,
    c;
    printf("Enter the numbers: ");
    scanf("%d %d %d %d", &n, &a,
    &b, &c);int ans = find_nth_term(n,
    a, b, c); printf("%d", ans);
    return 0;
}
```

Output 1:

Enter the numbers: 5 1 4 3
11

Output 2:

Enter the numbers: 7 8 9 6
128