C. Assignment 3.

Section-A

1. Accept 5 values and print them later on.

code:

```
#include <stdio.h>
int main() {
int values[5];
int i;
printf("Enter 5 integer values:\n");
for (int i=0; i<5; i++) {
  printf("Value %d: ", i+1);
  scanf("%d", &values[i]);
}
printf("The accepted values are:\n");
for (i=0; i<5; i++) {
  printf("%d\n", values[i]);
}
return 0;
}
```

Output:

```
Enter 5 integer values:
1: 2 4 3 5 7
2: 3: 4: 5: The accepted values are:
2
4
3
5
7
```

2. Accept 10 values and print 4th, 7th and 9th value.

code:
```
# include <stdio.h>
int main() {
int values [10];
int i;
printf ("Enter 10 integer values :\n");
for (i=0 ; i<10 ; i++) {
printf ("Enter value %d :", i+1);
scanf ("%d", &values [i]);
}
printf ("\n Printing specific values :\n");
printf ("4th value : %d\n", values [3]);
printf ("7th value : %d\n", values [6]);
printf ("9th value : %d\n", values [8]);
return 0;
}
```

Output:
```
Enter 10 integer values :
Enter value 1 : 1
Enter value 2 : 2
Enter value 3 : 3
Enter value 4 : 4
Enter value 5 : 5
Enter value 6 : 6
Enter value 7 : 7
Enter value 8 : 8
Enter value 9 : 9
Enter value 10 : 10
```

3. Accept 5 values and sort the array in ascending/descending order. Find out different techniques for sorting and identify the best one.

code:

```
#include <stdio.h>
void bubbleSort (int arr[], int n, int order) {
    int i, j, temp;
    for (i=0; i<n-1; i++) {
        if (order == 1) {
            if (arr[j] > arr[j+1]) {
                temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
            }
        }
    }
}
int main() {
    int arr[5];
    int i, order;
    printf ("Enter 5 integer values: \n");
    for (i=0; i<5, i++) {
        scanf ("%d", &arr[i]);
    }
    printf ("Enter 1 for ascending order, 0 for
            descending order: ");
    scanf ("%d", &order);

    bubbleSort (arr, 5, order);
```

```
printf ("Sorted array : ");
for (i=0 ; i<5 ; i++) {
    printf ("%d", arr [i]);
}
printf ("\n");
return 0;
}
```

Output:

```
Enter 5 integer values :
7 8 10 18 45
Enter 1 for ascending order, 0 for descending order : 0
Sorted array : 45 18 10 8 7
```

4. Print minimum number of notes required (1,2,5,10,20,50,100, 200, 500). Ex:- 1156 → 2×500, 1×200, 1×50, 1×5, 1×1.

Code:

```
#include <stdio.h>
int main() {
int amount = 1024;
int notes[] = {500, 200, 100, 50, 20, 10, 5, 2, 1};
int num_notes = sizeof(notes) / sizeof(notes[0]);
int count;
printf("For amount %d:\n", amount);
for(int i=0; i < num_notes; i++) {
    count = amount / notes[i];
    if (count > 0) {
        printf("%d x %d\n", count, notes[i]);
        amount %= notes[i];
    }
}
return 0;
}
```

Output:

For amount 1024:

2 x 500

1 x 20

2 x 2

5. Add two 2D arrays of same size and store the result in the 3rd one.

code:

```
#include <stdio.h>
#define ROWS 3
#define COLS 3
int main() {
    int array1 [ROWS][COLS] = {
        {1,2,3},
        {4,5,6},
        {7,8,9}
    };
    int array2 [ROWS] [COLS] = {
        {9,8,7},
        {6,5,4},
        {3,2,1}
    };
    int sum_array [ROWS] [COLS];
    for (int i=0; i<ROWS; i++) {
        for (int j=0; j<COLS; j++) {
            sum_array [i] [j] = array1 [i] [j] + array2 [i] [j];
        }
    }
    printf ("Sum of the two 2D arrays: \n");
    for (int i=0; i<ROWS; i++) {
        for (int j=0; j<COLS; j++) {
            printf ("%d", sum_array [i] [j]);
        }
        printf ("\n");
    }
    return 0;
}
```

Output: Sum of the two 2D arrays:

10 10 10

10 10 10

10 10 10

6. Multiply two 2D arrays and store the result in the 3rd one.

code:

```
#include <stdio.h>
#define R1 3
#define RC1 2
#define R2 2
#define C2 3
int main () {
  int matrix1 [R1][C1];
  int matrix2 [R2] [C2];
  int resultMatrix [R1] [C2];
  int i, j, k;
  printf ("Enter elements for first matrix (%d x %d): \n",
          R1, C1);
  for (i=0 ; i < R1 ; i++) {
    for (j=0 ; j < C1 ; j++) {
      printf (" Enter element matrix 1 [%d] [%d] :", i, j);
      scanf ("%d", & matrix1 [i] [j]);
    }
  }
  printf (" \nEnter elements for second matrix (%d x %d): \n",
          R2, C2);
  for (i=0; i<R2 ; i++) {
    for (j=0 ; j < C2 ; j++) {
      printf (" Enter element matrix2 [%d] [%d] : ", i, j);
      scanf ("%d", & Matrix2 [i] [j]);
    }
  }
```

```
for (i=0 ; i<R1 ; i++) {
    for (j=0; j<C2 ; j++) {
        result Matrix [i][j] = 0;
    }
}

for (i=0 ; i<R1 ; i++) {
    for (j=0 ; j<C2 ; j++) {
        for (k=0 ; k<C1 ; k++) {
            result Matrix [i][j] += matrix1 [i][k] * matrix2 [k][j];
        }
    }
}

printf ("\n Resultant matrix after multiplication
        (%d x %d) : \n", R1, C2 );
for (i=0 ; i<R1 ; i++) {
    for (j=0 ; j<C2 ; j++) {
        printf ("%d\t", result Matrix [i][j]);
    }
    printf ("\n");
}

return 0;
}
```

Output:

```
Enter elements for first matrix (3x2):
Enter element matrix1 [0][0] : 0
Enter element matrix1 [0][1] : 1
Enter element matrix1 [1][0] : 2
Enter element matrix1 [1][1] : 3
Enter element matrix1 [2][0] : 4
Enter element matrix1 [2][1] : 5
```

Enter elements for second matrix (2x3):
Enter element matrix2[0][0] : 6
Enter element matrix2[0][1] : 7
Enter element matrix2[0][2] : 8
Enter element matrix2[1][0] : 9
Enter element matrix2[1][1] : 10
Enter element matrix2[1][2] : 11

Resultant matrix after multiplication (3x3):

9	10	11
39	44	49
69	78	87

7. Obtain transpose of a 4x4 matrix.

code:

```
#include <stdio.h>
int main() {
    int matrix[4][4];
    int transpose[4][4];
    int i, j;
    printf("Enter elements of 4x4 matrix:\n");
    for (i=0; i<4; i++) {
        for (j=0; j<4; j++) {
            printf("Enter element matrix[%d][%d]:", i, j);
            scanf("%d", &matrix[i][j]);
        }
    }
    for (i=0; i<4; i++) {
        for (j=0; j<4; j++) {
            transpose[j][i] = matrix[i][j];
        }
    }
}
```

```
printf ("\n Transposed Matrix :\n");
for (i=0; i<4; i++) {
    for (j=0; j<4; j++) {
        printf ("%d\t", matrix[i][j]);
    }
    printf ("\n");
}

printf ("\n Transposed Matrix :\n");
for (i=0; i<4; i++) {
    for (j=0; j<4; j++) {
        printf ("%d\t", transpose[i][j]);
    }
    printf ("\n");
}

return 0;
}
```

Output:

```
Enter elements of 4x4 matrix:
Enter element matrix[0][0] : 0
Enter element matrix[0][1] : 1
Enter element matrix[0][2] : 2
Enter element matrix[0][3] : 3
Enter element matrix[1][0] : 4
Enter element matrix[1][1] : 5
Enter element matrix[1][2] : 6
Enter element matrix[1][3] : 7
Enter element matrix[2][0] : 8
Enter element matrix[2][1] : 9
Enter element matrix[2][2] : 10
Enter element matrix[2][3] : 11
Enter element matrix[3][0] : 12
```

Enter element matrix [3] [1] : 13
Enter element matrix [3] [2] : 14
Enter element matrix [3] [3] : 15

Original Matrix:

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Transposed Matrix:

0	4	8	12
1	5	9	13
2	6	10	14
3	7	11	15

8. Copy one array of 5 elements to another array of 10 elements (skipping one position).

Code:

```
#include <stdio.h>
int main() {
    int destinationArray[10] = {0};
    int j = 0;
    for (int i = 0; i < 5; i++) {
        if (j < 10) {
            destinationArray[j] = sourceArray[i];
            j += 2;
        }
    }
    printf("Source Array: ");
    for (int i = 0; i < 5; i++) {
        printf("%d", sourceArray[i]);
    }
    printf("\n");
    printf("Destination Array: ");
    for (int i = 0; i < 10; i++) {
        printf("%d", destinationArray[i]);
    }
    printf("\n");
    return 0;
}
```

Output:

```
Source Array: 10 20 30 40 50
Destination Array: 10 0 20 0 30 0 40 0 50 0
```

9. Reverse an array of maximum 5 elements.

Code:

```
#include <stdio.h>
int main() {
    int arr[5];
    int numElements;
    printf("Enter the number of elements (maximum 5): ");
    scanf("%d", &numElements);
    if (numElements > 5 || numElements < 0) {
    printf("Invalid number of elements. Please enter
            a number between 0 and 5.\n");
    return 1;
    }
    printf("Enter %d elements:\n", numElements);
    for (int i=0; i<numElements; i++) {
        printf("Elements %d: ", i+1);
        scanf("%d", &arr[i]);
    }
    printf("Original array: ");
    for (int i=0; i<numElements; i++) {
        printf("%d", arr[i]);
    }
    printf("\n");
    for (int i=0; i<numElements/2; i++) {
        int temp = arr[i];
        arr[i] = arr[numElements-1-i];
        arr[numElements-1-i] = temp;
    }
    printf("Reversed array:");
```

```
for (int i=0; i<numElements; i++) {
    printf("%d", arr[i]);
}
printf("\n");
return 0;
}
```

Output:
```
Enter the number of elements (maximum 5) : 3
Enter 3 elements :
Element 1 : 7
Element 2 : 18
Element 3 : 45
Original array : 7 18 45
Reversed array : 45 18 7
```

10. Find frequency of each number in an array (10 elements).

Code:
```
#include <stdio.h>
int main() {
    int arr[10] = {1, 2, 8, 3, 2, 2, 5, 1, 8, 8};
    int frequency[10];
    int visited = -1;
    for (int i=0; i<10; i++) {
        frequency[i] = 0;
    }
    for (int i = 0; i<10; i++) {
        int count = 1;
        for (int j = i+1; j<10; j++) {
            if (arr[i] == arr[j]) {
                count++;
                frequency[j] = visited;
            }
        }
```

```
        if (frequency[i] != visited) {
            frequency[i] = count;
        }
    }

    printf("Element | Frequency\n");
    printf("--------|----------\n");
    for (int i = 0; i < 10; i++) {
        if (frequency[i] != visited) {
            printf("%7d | %9d\n", arr[i], frequency[i]);
        }
    }

    return 0;
}
```

Output:

Element	Frequency
1	2
2	3
8	3
3	1
5	1

11. Shift all numbers by n positions within an array of 10 elements (left or right), pad with 0.

Code:

```
#include <stdio.h>
#include <string.h>
void shiftArray(int arr[], int size, int n, char
                direction) {
    int tempArr[size];
    memset(tempArr, 0, sizeof(tempArr));
```

```
    if (direction == 'R' || direction == 'r') {
        for (int i = 0; i < size; i++) {
            if (i - n >= 0) {
                tempArr[i] = arr[i - n];
            }
        }
    } else if (direction == 'L' || direction == 'l') {
        for (int i = 0; i < size; i++) {
            if (i + n < size) {
                tempArr[i] = arr[i + n];
            }
        }
    } else {
        printf("Invalid direction. Use 'L' for left or 'R'
                for right.\n");
        return;
    }
    for (int i = 0; i < size; i++) {
        arr[i] = tempArr[i];
    }
}
int main() {
    int arr[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
    int size = 10;
    int n = 3;
    char direction = 'R';
    printf("Original array: ");
    for (int i = 0; i < size; i++) {
        printf("%d", arr[i]);
    }
```

```
printf("\n");
shiftArray(arr, size, n, direction);
printf("\n");
shiftArray
printf("Array after %d position %s shift:", n,
        (direction == 'R' || direction == 'r') ?
        "right" : "left");
for(int i = 0; i < size; i++) {
    printf("%d", arr[i]);
}
printf("\n");
int arr2[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
direction = 'L';
n = 2;
printf("Original array 2:");
for(int i = 0; i < size; i++) {
    printf("%d", arr2[i]);
}
printf("\n");
shiftArray(arr2, size, n, direction);
printf("Array 2 after %d position %s shift:", n,
        (direction == 'R' || direction == 'r') ? "right" : "left");
for(int i = 0; i < size; i++) {
    printf("%d", arr2[i]);
}
printf("\n");
return 0;
}
```

Output: Original array: 1 2 3 4 5 6 7 8 9 10
Array after 3 position right shift: 0 0 0 1 2 3 4 5 6 7
Original array2: 1 2 3 4 5 6 7 8 9 10
Array 2 after 2 position left shift:
3 4 5 6 7 8 9 10 0 0

12. Insert a new number at the beginning of the array.

Code:
```
#include <stdio.h>
int main() {
  int arr[100] = {1, 2, 3, 4, 5};
  int size = 5;
  int new_number = 0;
  printf("Original array:");
  for (int i=0; i<size; i++) {
    printf("%d", arr[i]);
  }
  printf("\n");
  for (int i = size - 1; i >= 0; i--) {
    arr[i+1] = arr[i];
  }
  arr[0] = new_number;
  size ++;
  printf("Array after insertion:");
  for (int i=0; i<size; i++) {
    printf("%d", arr[i]);
  }
  printf("\n");
  return 0;
}
```

Output: Original array: 1 2 3 4 5
Array after insertion: 0 1 2 3 4 5

13. Insert a new number at a particular position in an array.

Code:

```
#include <stdio.h>
int main() {
    int array[100], position, c, n, value;
    printf("Enter number of elements in array \n");
    scanf("%d", &n);
    printf("Enter %d elements \n", n);
    for (c=0; c<n; c++)
        scanf("%d", &array[c]);
    printf("Enter the location where you wish to insert
            an element \n");
    scanf("%d", &position);
    printf("Enter the value to insert \n");
    scanf("%d", &value);
    for (c= n-1; c>=position -1; c--)
        array[c+1] = array[c];
    array[position -1] = value;
    printf("Resultant array is \n");
    for (c=0; c<=n; c++)
        printf("%d \n", array[c]);
    return 0;
}
```

Output:

```
Enter number of elements in array : 1
Enter 1 elements : 7
Enter the location where you wish to insert an element : 1
Enter the value to insert : 8
Resultant array is
8
7
```

14. Insert a new number at the last position of an array.

Code:
```
#include <stdio.h>
int main() {
    int arr[100] = {1, 2, 3, 4, 5};
    int size = 5;
    int new_number = 6;
    if (size < 100) {
        arr[size] = new_number;
        size++;
        printf("Array after insertion:");
        for (int i=0; i<size; i++) {
            printf("%d", arr[i]);
        }
        printf("\n");
    } else {
        printf("Array is full. Cannot insert new number.
                \n");
    }
    return 0;
}
```

Output: Array after insertion: 1 2 3 4 5 6

15. Delete a value from the first position of an array.

Code:
```
#include <stdio.h>
int main() {
    int arr[100];
    int size, i;
    printf("Enter the size of the array:");
    scanf("%d", &size);
```

```
printf ("Enter the elements of the array :");
for (i=0 ; i<size; i++) {
    scanf ("%d", &arr[i]);
}
for (i=0; i<size-1; i++) {
    arr[i] = arr[i+1];
}
size--;
printf ("Array after deleting the first element :");
for (i=0; i<size; i++) {
    printf ("%d", arr[i]);
}
printf ("\n");
return 0;
}
```

Output: Element the size of the array : 2
Eter the elements of the array : 12 7
Array after deleting the first element : 7

16. Delete a value from a particular position in an array.

Code:
```
#include <stdio.h>
int main() {
    int arr[200];
    int size, i, pos;
    printf ("Enter the size of the array :");
    scanf ("%d", &size);
    printf ("Enter %d elements in the array :", size);
    for (i=0 ; i<size ; i++) {
        scanf ("%d", &arr[i]);
    }
```

```
printf("Enter the position of the element to delete: ");
scanf("%d", &pos);
if (pos < 0 || pos > size) {
    printf("Invalid position!\n");
} else {
    for (i = pos - 1; i < size - 1; i++) {
        arr[i] = arr[i+1];
    }
    size--;
    printf("Array after deletion: ");
    for (i=0; i < size; i++) {
        printf("%d", arr[i]);
    }
    printf("\n");
    return 0;
}
```

Output:

```
Enter the size of the array: 2
Enter 2 elements in the array: 18 7
Enter the position of the element to delete: 1
Array after deletion: 7.
```

17. Delete a value from the last position of an array.

Code:
```
#include <stdio.h>
int main() {
    int arr[] = {10, 20, 30, 40, 50};
    int size = 5;
    int i;
    printf("Original array :");
    for (i = 0; i < size; i++) {
        printf("%d", arr[i]);
    }
    printf("\n");
    if (size > 0) {
        size--;
    }
    printf("Array after deleting the last element :");
    for (i = 0; i < size; i++) {
        printf("%d", arr[i]);
    }
    printf("\n");
    return 0;
}
```

Output: Original array : 10 20 30 40 50.
Array after deleting the last element: 10 20 30 40.

18. Delete a value from the array. (search and remove).

Code:
```
#include <stdio.h>
int deleteElement(int arr[], int size, int value_to_delete) {
    int i, j;
    int found = 0;
```

```
    for (i=0; i<size; i++) {
        if (arr[i] == value_to_delete) {
            found = 1;
            break;
        }
    }
    if (found) {
        for (j=i; j<size-1; j++) {
            arr[j] = arr[j+1];
        }
        printf("Element %d deleted successfully.\n", value_to_
                delete);
        return size-1;
    } else {
        printf("Element %d not found in the array.\n",
                value_to_delete);
        return size;
    }
}
int main() {
    int arr[] = {10,20,30,40,50};
    int size = sizeof(arr)/sizeof(arr[0]);
    int value_to_delete = 30;
    printf("Original array: ");
    for (int i=0; i<size; i++) {
        printf("%d", arr[i]);
    }
    printf("\n");
    size = deleteElement(arr, size, value_to_delete);
```

```
printf ("Array after deletion:");
for (int i=0; i<size; i++) {
    printf ("%d", arr[i]);
}
printf ("\n");
return 0;
}
```

Output: Original array : 10 20 30 40 50.
Element 30 deleted successfully.
Array after deletion : 10 20 40 50.

19. Search a value within an array.

Code:

```
#include <stdio.h>
int main() {
    int arr[] = {10, 20, 30, 40, 50};
    int n = sizeof(arr) / sizeof(arr[0]);
    int searchValue = 30;
    int found = 0;
    printf ("Searching for %d in the array...\n",
            searchValue);
    for (int i=0; i<n; i++) {
        if (arr[i] == searchValue) {
            printf ("Value %d found at index %d.\n",
                    searchValue, i);
            found = 1;
            break;
        }
    }
    if (found == 0) {
        printf ("Value %d not found in the array.\n",
                searchvalue);
    }
```

```
    return 0;
}
```

Output:
```
Searching for 30 in the array...
Value 30 found at index 2.
```

Section - B

1. Find out length of a string.

Code:
```
#include <stdio.h>
#include <string.h>
int main() {
    char str[] = "Hello, World!";
    int length;
    length = strlen(str);
    printf("The length of the string is: %d\n", length);
    return 0;
}
```

Output:
```
The length of the string is: 13
```

2. Convert a string to lowercase.

Code:
```
#include <stdio.h>
#include <ctype.h>
int main() {
    char str[] = "HeLlO WoRlD";
    printf("Original string: %s\n", str);
    for (int i=0; str[i] != '\0'; i++) {
        str[i] = tolower(str[i]);
    }
    printf("Lowercase string: %s\n", str);
    return 0;
}
```

Output:
```
Original string:
HeLlO WoRlD
Lowercase string:
hello world
```

3. Convert a string to uppercase.

Code:
```
#include <stdio.h>
#include <ctype.h>
void to_uppercase (char *str) {
    int i = 0;
    while (str[i] != '\0') {
        str[i] = toupper (str[i]);
        i++;
    }
}
int main() {
    char myString[100];
    printf ("Enter a string: ");
    scanf ("%s", myString);
    to_uppercase (myString);
    printf ("The uppercase string is: %s\n", myString);
    return 0;
}
```

Output:
```
Enter a string: hello.
The uppercase string is: HELLO.
```

4. Convert a string to toggle case. (eg. Ab(→ aBc)

Code:
```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
void toggleCase (char *str) {
    int i = 0;
    while (str[i] != '\0') {
        if (islower (str[i])) {
            str[i] = toupper (str[i]);
```

```
        } else if (isupper(str[i])) {
            str[i] = tolower(str[i]);
        }
        i++;
    }
}
int main() {
    char str[100];
    printf("Enter a string:");
    gets(str);
    toggleCase(str);
    printf("Toggled case string: %s\n", str);
    return 0;
}
```

Output:
```
Enter a string: Darshan.
Toggled case string: dARSHAN.
```

5. Copy one string to another.

Code:
```
#include <stdio.h>
#include <string.h>
int main() {
    char source_string[] = "Hello, World!";
    char destination_string[20];
    strcpy(destination_string, source_string);
    printf("Source string: %s\n", source_string);
    printf("Destination string: %s\n", destination_string);
    return 0;
}
```

Output:
```
Source string: Hello, World!
Destination string: Hello, World!
```

6. Compare two strings lexicographically and print which one is greater, smaller, or same.

Code:
```
#include <stdio.h>
#include <string.h>
int main() {
    char str1[100];
    char str2[100];
    printf("Enter the first string:");
    scanf("%s", str1);
    printf("Enter the second string:");
    scanf("%s", str2);
    int result = strcmp(str1, str2);
    if (result > 0) {
        printf("The first string is greater than the second string.\n");
    } else if (result < 0) {
        printf("The first string is smaller than the second string.\n");
    } else {
        printf("The two strings are the same.\n");
    }
    return 0;
}
```

Output:
```
Enter the first string: 7
Enter the second string: 18
The first string is greater than the second string.
```

7. Reverse a string.

Code:
```
#include <stdio.h>
#include <string.h>
int main() {
    char str[100];
```

```
    char temp;
    int i, j;
    printf("Enter a string:");
    scanf("%s", str);
    j = strlen(str) - 1;
    for(i=0; i<j; i++, j--){
        temp = str[i];
        str[i] = str[j];
        str[j] = temp;
    }
    printf("Reversed string is: %s\n", str);
    return 0;
}
```

Output:
```
Enter a string: 45
Reversed string is: 54
```

8. Check whether a string is a Palindrome.

Code:
```
#include <stdio.h>
#include <string.h>
int main(){
    char str[100];
    int i, length;
    int is_palindrome = 1;
    printf("Enter a string:");
    scanf("%s", str);
    length = strlen(str);
    for(i=0; i<length/2; i++){
        if(str[i] != str[length-1-i]){
            is_palindrome = 0;
            break;
        }
    }
```

```
    if (is_palindrome) {
        printf ("%s is a palindrome.\n", str);
    } else {
        printf ("%s is not a palindrome.\n", str);
    }

    return 0;
}
```

Output: Enter a string : 7

7 is a palindrome.

9. Concatenate one string at the end of another string.

Code:
```
#include <stdio.h>
#include <string.h>
int main() {
    char str1[100] = "Hello, ";
    char str2[] = "world!";
    strcat(str1, str2);
    printf("The concatenated string is: %s\n", str1);
    return 0;
}
```

Output: The concatenated string is: Hello, world!

10. Print characters of a string vertically.

Code:
```
#include <stdio.h>
#include <string.h>
int main() {
    char str[100];
    printf("Enter a string:");
    scanf("%s", str);
    for (int i=0; str[i] != '\0'; i++) {
        printf("%c\n", str[i]);
    }
    return 0;
}
```

Output: Enter a string: 77

7

7

11. Print reversed string vertically character by character.

Code:

```
#include <stdio.h>
#include <string.h>
int main() {
    char str[100];
    int length;
    printf("Enter a string:");
    scanf("%s", str);
    length = strlen(str);
    for(int i=length-1; i>=0; i--) {
        printf("%c\n", str[i]);
    }
    return 0;
}
```

Output:

```
Enter a string : 18
8
1
```

Page No.
Date: / /

12. Print frequency of each vowel in a given string.

Code:

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
int main() {
    char str[100];
    int vowel_count[5] = {0};
    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);
    for (int i=0; str[i] != '\0'; i++) {
        char ch = tolower(str[i]);
        switch (ch) {
            case 'a':
                vowel_count[0]++;
                break;
            case 'e':
                vowel_count[1]++;
                break;
            case 'i':
                vowel_count[2]++;
                break;
            case 'o':
                vowel_count[3]++;
                break;
            case 'u':
                vowel_count[4]++;
                break;
        }
}
```

```
printf("Frequency of each vowel : \n");
printf("a : %d \n", vowel_count[0]);
printf("e : %d \n", vowel_count[1]);
printf("i : %d \n", vowel_count[2]);
printf("o : %d \n", vowel_count[3]);
printf("u : %d \n", vowel_count[4]);
return 0;
}
```

Output: Enter a string : darshan

Frequency of each vowel:

a : 2

e : 0

i : 0

o : 0

u : 0