Q1:

#include <stdio.h>

#include <stdlib.h>

*// declare functions*

void **printArray**(int array[], int len);

void **generateArray**(int array[], int len);

void **insertionSort**(int array[], int len);

void **initArray**(int array[], int len, int value);

int **findDuplicates**(int array[], int track[], int len);

int **main**()

{

*// declare variables*

    int total = 0;

    int n = 100;

    int \*arr = (int \*)**malloc**(sizeof(int) \* n);

    int \*track = (int \*)**malloc**(sizeof(int) \* n);

*// generating array with random numbers*

**generateArray**(arr, n);

*// printing array*

**printf**("\nElements:\n");

**printArray**(arr, n);

*// sorting the array*

**insertionSort**(arr, n);

*// intializing track array*

**initArray**(track, n, 1);

*// finding duplicates from sorted array*

    total = **findDuplicates**(arr, track, n);

*// printing duplicates*

**printf**("\n\nDuplicates:\n");

    for (int i = 0; i < n; i++)

        if (track[i] > 1)

**printf**("%d - %d\n", arr[i], track[i]);

**printf**("\nTotal: %d\n", total);

*// free memory*

**free**(arr);

**free**(track);

    return 0;

}

*// function to generate random integer array*

void **generateArray**(int array[], int len)

{

**srand**(**time**(**NULL**));

    for (int i = 0; i < len; i++)

        array[i] = **rand**() % 100;

}

*// print array*

void **printArray**(int array[], int len)

{

    for (int i = 0; i < len; i++)

**printf**("%d ", array[i]);

}

*// initialize array*

void **initArray**(int array[], int len, int value)

{

    for (int i = 0; i < len; i++)

        array[i] = value;

}

*// insersion sort an array*

void **insertionSort**(int array[], int len)

{

    for (int i = 1; i < len; i++)

    {

        int temp = array[i];

        int j = i - 1;

        while (j >= 0 && array[j] > temp)

        {

            array[j + 1] = array[j];

            j--;

        }

        array[j + 1] = temp;

    }

}

*// find duplicates from a sorted array*

int **findDuplicates**(int array[], int track[], int len)

{

    int total = 0;

    for (int i = 0; i < len; i++)

    {

        for (int j = i + 1; j < len; j++)

        {

            if (array[i] == array[j])

            {

                track[i]++;

                total++;

            }

            else

            {

                i = j;

                break;

            }

        }

    }

    return total;

}

Q2:

#include <stdio.h>

#include <stdlib.h>

*// declare functions*

void **printArray**(int array[], int len);

void **generateArray**(int array[], int len);

int **removeDuplicates**(int array[], int len);

void **leftShift**(int array[], int currIndex, int len);

int **main**()

{

*// declare variables*

    int total = 0;

    int n;

    int \*arr;

*// input array length*

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

**scanf**("%d", &n);

*// allocate memory for array*

    arr = (int \*)**malloc**(sizeof(int) \* n);

*// input array*

    for (int i = 0; i < n; i++)

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

*// generating array with random numbers*

*// generateArray(arr, n); // for testing*

*// printing array*

**printf**("\nElements (Before Remove):\n");

**printArray**(arr, n);

*// remove duplicates*

    total = **removeDuplicates**(arr, n);

*// printing array*

**printf**("\n\nElements (After Remove):\n");

**printArray**(arr, total);

*// free memory*

**free**(arr);

    return 0;

}

*// function to generate random integer array*

void **generateArray**(int array[], int len)

{

**srand**(**time**(**NULL**));

    for (int i = 0; i < len; i++)

        array[i] = **rand**() % 100;

}

*// print array*

void **printArray**(int array[], int len)

{

    for (int i = 0; i < len; i++)

**printf**("%d ", array[i]);

}

*// remove current element and shift left*

void **leftShift**(int array[], int currIndex, int len)

{

    while (currIndex < len - 1)

    {

        array[currIndex] = array[currIndex + 1];

        currIndex++;

    }

}

*// remove duplicates*

int **removeDuplicates**(int array[], int len)

{

    for (int i = 0; i < len; i++)

    {

        for (int j = i + 1; j < len; j++)

        {

            if (array[i] == array[j])

            {

**leftShift**(array, j, len);

                len--;

            }

        }

    }

    return len;

}

Q3:

#include <stdio.h>

#include <stdlib.h>

void **printArray**(int array[], int len);

void **generateArray**(int array[], int len);

int **binarySearch**(int array[], int len, int key);

void **intersection**(int array1[], int len1, int array2[], int len2);

int **main**()

{

    int n = 100;

    int \*arr1, \*arr2;

*// allocate memory for arrays*

    arr1 = (int \*)**malloc**(sizeof(int) \* n);

    arr2 = (int \*)**malloc**(sizeof(int) \* n);

*// generating array with random numbers*

**printf**("\nGenerating array 1...\n");

**generateArray**(arr1, n);

**printArray**(arr1, n);

**printf**("\n\nGenerating array 2...\n");

**generateArray**(arr2, n);

**printArray**(arr2, n);

*// intersect & print*

**intersection**(arr1, n, arr2, n);

*// free memory*

**free**(arr1);

**free**(arr2);

    return 0;

}

*// function to generate random integer array*

void **generateArray**(int array[], int len)

{

**srand**(**time**(**NULL**));

    for (int i = 0; i < len; i++)

        array[i] = **rand**() % 100;

}

*// print array*

void **printArray**(int array[], int len)

{

    for (int i = 0; i < len; i++)

**printf**("%d ", array[i]);

}

*// intersection of two sorted arrays*

void **intersection**(int array1[], int len1, int array2[], int len2)

{

    int i = 0, j = 0;

**printf**("\n\nIntersection of two sorted arrays:\n");

    while (i < len1 && j < len2)

    {

        if (array1[i] == array2[j])

        {

**printf**("%d ", array1[i]);

            i++;

            j++;

        }

        else if (array1[i] < array2[j])

        {

            i++;

        }

        else

        {

            j++;

        }

    }

}

Q4:

#include <stdio.h>

#include <stdlib.h>

void **printArray**(int array[], int len);

void **generateArray**(int array[], int len);

int **findSmallest**(int array[], int len);

int **findSmallestGreaterThanK**(int array[], int len, int k);

int **main**()

{

    int k = 0, num;

    int n = 10;

    int \*arr = (int \*)**malloc**(sizeof(int) \* n);

**generateArray**(arr, n);

**printf**("\nArray Elements: ");

**printArray**(arr, n);

**printf**("\n\nEnter the index number: ");

**scanf**("%d", &k);

    num = **findSmallest**(arr, n);

    for (int i = 1; i < k; i++)

    {

        num = **findSmallestGreaterThanK**(arr, n, num);

    }

**printf**("\n%d Smallest number is %d", k, num);

**free**(arr);

    return 0;

}

*// function to generate random integer array*

void **generateArray**(int array[], int len)

{

**srand**(**time**(**NULL**));

    for (int i = 0; i < len; i++)

        array[i] = **rand**() % 100;

}

*// print array*

void **printArray**(int array[], int len)

{

    for (int i = 0; i < len; i++)

**printf**("%d ", array[i]);

}

*// find smallest element in unsorted array*

int **findSmallest**(int array[], int len)

{

    int smallest = array[0];

    for (int i = 1; i < len; i++)

    {

        if (array[i] < smallest)

            smallest = array[i];

    }

    return smallest;

}

*// find smallest greater than k*

int **findSmallestGreaterThanK**(int array[], int len, int k)

{

    int smallest = array[0];

    for (int i = 1; i < len; i++)

    {

        if (array[i] > k && array[i] < smallest)

            smallest = array[i];

    }

    return smallest;

}

Q5:

#include <stdio.h>

#include <stdlib.h>

void **printArray**(int array[], int len);

void **generateArray**(int array[], int len);

void **reverseArray**(int array[], int len);

int **main**()

{

    int n = 10;

    int \*arr = (int \*)**malloc**(sizeof(int) \* n);

**generateArray**(arr, n);

**printf**("\nArray Elements: ");

**printArray**(arr, n);

*// reverse the array*

**reverseArray**(arr, n);

*// print the array*

**printf**("\n\nReversed Array: ");

**printArray**(arr, n);

**free**(arr);

    return 0;

}

*// reverse array*

void **reverseArray**(int array[], int len)

{

    int i = 0;

    while (i < len / 2)

    {

        int tmp = array[i];

        array[i] = array[len - i - 1];

        array[len - i - 1] = tmp;

        i++;

    }

}

*// function to generate random integer array*

void **generateArray**(int array[], int len)

{

**srand**(**time**(**NULL**));

    for (int i = 0; i < len; i++)

        array[i] = **rand**() % 100;

}

*// print array*

void **printArray**(int array[], int len)

{

    for (int i = 0; i < len; i++)

**printf**("%d ", array[i]);

}

Q6:

#include <stdio.h>

#include <stdlib.h>

void **generateArray**(int[], int);

void **swap**(int \*, int \*);

void **bubbleSort**(int[], int);

void **readIntArrayFromFile**(int[], char[]);

void **writeIntArrayToFile**(int[], int, char[]);

int **main**()

{

    int n = 500;

    int \*arr = (int \*)**malloc**(sizeof(char) \* n);

**generateArray**(arr, 500);

**writeIntArrayToFile**(arr, 500, "in.txt");

**readIntArrayFromFile**(arr, "in.txt");

**bubbleSort**(arr, 500);

**writeIntArrayToFile**(arr, 500, "out.txt");

**free**(arr);

    return 0;

}

*// generate random integer numbers in the range from -249 to 250*

void **generateArray**(int arr[], int size)

{

    int i;

**srand**(**time**(**NULL**));

    for (i = 0; i < size; i++)

    {

*// formula: rand() % (max\_number + 1 - minimum\_number) + minimum\_number*

        arr[i] = **rand**() % (250 + 1 + 249) - 249;

    }

}

*// function to swap two elements using pointer*

void **swap**(int \*a, int \*b)

{

    int temp = \*a;

    \*a = \*b;

    \*b = temp;

}

*//function to sort a int array using bubble sort*

void **bubbleSort**(int arr[], int n)

{

    int i, j;

    for (i = 0; i < n; i++)

    {

        for (j = 0; j < n - 1; j++)

        {

            if (arr[j] > arr[j + 1])

            {

**swap**(&arr[j], &arr[j + 1]);

            }

        }

    }

}

*// read int from a text file and store it in an array*

void **readIntArrayFromFile**(int arr[], char fileName[])

{

    int i = 0;

**FILE** \*fp; *// File pointer*

    fp = **fopen**(fileName, "r");

    if (fp == **NULL**)

    {

**printf**("File not found\n");

**exit**(0);

    }

    while (!**feof**(fp))

    {

**fscanf**(fp, "%d", &arr[i]);

        i++;

    }

**fclose**(fp);

}

*// write int array to a text file*

void **writeIntArrayToFile**(int arr[], int size, char fileName[])

{

    int i;

**FILE** \*fp; *// File pointer*

    fp = **fopen**(fileName, "w");

    if (fp == **NULL**)

    {

**printf**("File not found\n");

**exit**(0);

    }

    for (i = 0; i < size; i++)

    {

**fprintf**(fp, "%d\n", arr[i]);

    }

**fclose**(fp);

}

Q7:

#include <stdio.h>

#include <stdlib.h>

void **generateArray**(char arr[], int size);

void **insertionSort**(char arr[], int n);

void **readIntArrayFromFile**(char arr[], char fileName[]);

void **writeIntArrayToFile**(char arr[], char fileName[]);

void **printCharArrayToConsole**(char arr[]);

int **main**()

{

    int n = 1001;

    char \*arr = (char \*)**malloc**(sizeof(char) \* n);

**generateArray**(arr, 1000);

**writeIntArrayToFile**(arr, "in.txt");

**readIntArrayFromFile**(arr, "in.txt");

**insertionSort**(arr, 1000);

**printCharArrayToConsole**(arr);

**free**(arr);

    return 0;

}

*// generate random upper case characters*

void **generateArray**(char arr[], int size)

{

    int i;

**srand**(**time**(**NULL**));

    for (i = 0; i < size; i++)

    {

        arr[i] = (char)(**rand**() % 26 + 65);

    }

    arr[i] = '\0';

}

*//function to sort a char array using insertion sort*

void **insertionSort**(char arr[], int n)

{

    for (int i = 1; i < n; i++)

    {

*// Storing current element & previous index*

        int current = arr[i];

        int j = i - 1;

*// shifting previous element(s) to right if it's bigger than current*

        while (j > -1 && arr[j] > current)

        {

            arr[j + 1] = arr[j];

            --j;

        }

*// placing current element to right position*

        arr[j + 1] = current;

    }

}

*// read char from a text file and store it in an array*

void **readIntArrayFromFile**(char arr[], char fileName[])

{

    int i = 0;

**FILE** \*fp; *// File pointer*

    fp = **fopen**(fileName, "r");

    if (fp == **NULL**)

    {

**printf**("File not found\n");

**exit**(0);

    }

    while (!**feof**(fp))

    {

**fscanf**(fp, "%c", &arr[i]);

        i++;

    }

    arr[i] = '\0';

**fclose**(fp);

}

*// write char array to a text file*

void **writeIntArrayToFile**(char arr[], char fileName[])

{

**FILE** \*fp; *// File pointer*

    fp = **fopen**(fileName, "w");

    while (\*arr != '\0')

    {

**fprintf**(fp, "%c", \*arr);

        arr++;

    }

**fclose**(fp);

}

*// print out the array to console*

void **printCharArrayToConsole**(char arr[])

{

    int i = 0;

    while (arr[i] != '\0')

    {

**printf**("%c", arr[i]);

        i++;

    }

}

Q8:

#include <stdio.h>

#include <stdlib.h>

void **printArray**(int array[], int len);

void **generateArray**(int array[], int len);

void **replace**(int array[], int len);

int **main**()

{

    int n = 10;

    int \*arr = (int \*)**malloc**(sizeof(int) \* n);

**generateArray**(arr, n);

**printf**("\nArray Elements: ");

**printArray**(arr, n);

*// replace*

**replace**(arr, n);

*// print array*

**printf**("\n\nReplaced Array: ");

**printArray**(arr, n);

**free**(arr);

    return 0;

}

*// find divisible by 3 and replace by -1*

void **replace**(int array[], int len)

{

    for (int i = 0; i < len; i++)

    {

        if (array[i] % 3 == 0)

            array[i] = -1;

    }

}

*// function to generate random integer array*

void **generateArray**(int array[], int len)

{

**srand**(**time**(**NULL**));

    for (int i = 0; i < len; i++)

        array[i] = **rand**() % 100;

}

*// print array*

void **printArray**(int array[], int len)

{

    for (int i = 0; i < len; i++)

**printf**("%d ", array[i]);

}

Q9:

#include <stdio.h>

#include <stdlib.h>

void **inputIntToArray**(int[], int);

void **insertionSortDescending**(int[], int);

int **sumArrayFromIndexToIndex**(int[], int, int);

int **main**()

{

    int n;

    int i;

    int sum\_i = 0;

    int sum\_j = 0;

**scanf**("%d", &n);

    int \*arr = (int \*)**malloc**(n \* sizeof(int));

**inputIntToArray**(arr, n);

**insertionSortDescending**(arr, n);

    for (i = 0; i < n; i++)

    {

        sum\_i += arr[i];

        sum\_j = **sumArrayFromIndexToIndex**(arr, i + 1, n - 1);

        if (sum\_i > sum\_j)

        {

**printf**("%d\n", i + 1);

            break;

        }

    }

**free**(arr);

    return 0;

}

*// take int input to an array*

void **inputIntToArray**(int arr[], int size)

{

    int i;

    for (i = 0; i < size; i++)

    {

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

    }

}

*// sort an array decending using insertion sort*

void **insertionSortDescending**(int arr[], int n)

{

    for (int i = 1; i < n; i++)

    {

*// Storing current element & previous index*

        int current = arr[i];

        int j = i - 1;

*// shifting previous element(s) to right if it's smaller than current*

        while (j > -1 && arr[j] < current)

        {

            arr[j + 1] = arr[j];

            --j;

        }

*// placing current element to right position*

        arr[j + 1] = current;

    }

}

*// sum of the array from index n to index m*

int **sumArrayFromIndexToIndex**(int arr[], int n, int m)

{

    int sum = 0;

    int i;

    for (i = n; i <= m; i++)

    {

        sum += arr[i];

    }

    return sum;

}

Q10:

#include <stdio.h>

#include <stdlib.h>

void **generateArray**(int array[], int len);

void **printArray**(int array[], int len);

int **main**()

{

    int index1 = 0, index2 = 1;

    unsigned int diff, n;

*// scanf("%d", &n); // testing*

    n = 10;

    int \*arr = (int \*)**malloc**(sizeof(int) \* n);

**generateArray**(arr, n);

*// for (int i = 0; i < n; i++) // testing*

*//     scanf("%d", &arr[i]);*

**printf**("\nArray Elements: ");

**printArray**(arr, n);

    diff = arr[0] - arr[1];

    for (int i = 0; i < n; i++)

    {

        int tmp;

        for (int j = 0; j < n; j++)

        {

            if (i == j)

                continue; *// always gives 0, so ignored*

            tmp = (arr[i]) - (arr[j]);

            if (tmp < diff)

            {

                diff = tmp;

                index1 = i;

                index2 = j;

            }

        }

    }

**printf**("\n\n%d - %d = %d\n", arr[index1], arr[index2], diff);

**free**(arr);

    return 0;

}

*// function to generate random integer array*

void **generateArray**(int array[], int len)

{

**srand**(**time**(**NULL**));

    for (int i = 0; i < len; i++)

        array[i] = **rand**() % 100;

}

*// print array*

void **printArray**(int array[], int len)

{

    for (int i = 0; i < len; i++)

**printf**("%d ", array[i]);

}