

PPS5

Q1

Aim:

Write a 'C' program to sort the array of n elements using Selection sort. Get the user input for 'n'.

Procedure:**Input:**

Number of elements, n

Next n lines contain n numbers

Output:

Sorted Array

Algorithm:

Step 1: Read n

Step 2: Initialise array of size n

Step 3: Use for loop to read elements in the array

Step 4: Selection Sort Algorithm:

Step A: Iterate through n-1 elements using counter i

Step B: For each iteration set min to ith element of the array

Step C: Check if current element is greater than subsequent elements

If yes, set min to the subsequent element and store the index of that element

Step D: If min is less than initial element, swap the elements

Step 5: Display sorted Array

Code:

```
#include <stdio.h>

int main() {

    // Initialise variables and Read n
    int n, i, j, k, t, x, y, min, swap;
    printf("Number of elements in the array: ");
    scanf("%d",&n);

    // Initialise array and read elements of array
    int arr[n];
    for (x = 0; x < n; x++) {
        printf("Enter element %d: ", x+1);
        scanf("%d",&arr[x]);
    }
}
```

```

// Selection Sort
for (i = 0; i < n - 1; i++) {
    min = arr[i];
    swap = 0;
    for (j = i + 1; j < n; j++) {

        if (min > arr[j]) {
            swap = 1;
            min = arr[j];
            k = j;
        }

    }

    if (swap) {
        t = arr[i];
        arr[i] = min;
        arr[k] = t;
    }
}

// Print sorted array
printf("Sorted Array: ");
for (y = 0; y < n; y++) {
    printf("%d ", arr[y]);
}
}

```

The screenshot shows a Visual Studio Code editor with a C++ program for Selection Sort. The code is in a file named 'selectionsort.c' and includes comments for each step: reading n, reading the array, performing the selection sort, and printing the sorted array. The terminal window shows the execution output, including the number of elements (5) and the sorted array (1 2 3 4 5).

```

C:\selectionsort\selectionsort.c:1 X
C:\selectionsort\selectionsort.c:1 main()
1 #include <stdio.h>
2
3 int main() {
4
5     // Initialize variables and Read n
6     int n, i, j, k, t, x, y, min, swap;
7     printf("Number of elements in the array: ");
8     scanf("%d", &n);
9
10    // Initialize array and read elements of array
11    int arr[n];
12    for (x = 0; x < n; x++) {
13        printf("Enter element %d: ", x+1);
14        scanf("%d", &arr[x]);
15    }
16
17    // Selection Sort
18    for (i = 0; i < n - 1; i++) {
19        min = arr[i];
20        swap = 0;
21        for (j = i + 1; j < n; j++) {
22
23            if (min > arr[j]) {
24                swap = 1;
25                min = arr[j];
26                k = j;
27            }
28        }
29
30        if (swap) {
31            t = arr[i];
32            arr[i] = min;
33            arr[k] = t;
34        }
35    }
36
37    // Print sorted array
38    printf("Sorted Array: ");
39    for (y = 0; y < n; y++) {
40        printf("%d ", arr[y]);
41    }
42
43 }

```

Terminal Output:

```

PS C:\Users\Vidhi Shah\Desktop\VIITC Sem 2\C++ Lab\PPSS> gcc -o selectionsort selectionsort.c
PS C:\Users\Vidhi Shah\Desktop\VIITC Sem 2\C++ Lab\PPSS> ./selectionsort
Number of elements in the array: 5
Enter element 1: 2
Enter element 2: 4
Enter element 3: 3
Enter element 4: 1
Enter element 5: 5
Sorted Array: 1 2 3 4 5
PS C:\Users\Vidhi Shah\Desktop\VIITC Sem 2\C++ Lab\PPSS>

```

Q2

Aim:

Write a 'C' program to eliminate the duplicate elements from an array of n elements. Get the user input for 'n'.

Procedure:

Input:

Number of elements, n

Next n lines contain n numbers

Output:

Array with no duplicate elements

Algorithm:

Step 1: Read n

Step 2: Initialise original array (arr) and new array (arr1) of size n

Step 3: Use for loop to read elements in the arr

Step 4: Elimination of duplicate elements algorithm:

Step A: Assign first element of arr as first element of arr1, set k to 1

Step B: Iterate through n elements of arr

Step C: For each iteration set add to 1

Step D: Check if current element of arr is equal to any element of arr1

If yes, set add to 0

Step E: If add is 1, add the current element of arr to arr 1 at kth position, k = k + 1

Step 5: Display new array with no duplicate elements

Code:

```
int main() {

    // Initialise variables and Read n
    int n, i, j, k = 0, x, y, add;
    printf("Number of elements in the array: ");
    scanf("%d",&n);

    // Initialise array and read elements of array
    int arr[n], arr1[n];
    for (x = 0; x < n; x++) {
        printf("Enter element %d: ", x+1);
        scanf("%d",&arr[x]);
    }

    // Store unique elements from original array in new array
    arr1[0] = arr[0]; //1
    k = 1;
```

```

for (i = 0; i < n; i++) {
    add = 1;

    for (j = 0; j < k; j++) {
        if (arr[i] == arr1[j]) {
            add = 0;
        }
    }

    if (add) {
        arr1[k] = arr[i];
        k = k + 1;
    }
}

// Print array with no duplicate elements
printf("Final Array: ");
for (y = 0; y < k; y++) {
    printf("%d ",arr1[y]);
}
}

```

The screenshot shows the Visual Studio Code interface with the following components:

- Editor:** Displays the C program 'elmnduplicate.c'. The code includes headers, reads input, and implements a logic to store unique elements in a new array.
- Terminal:** Shows the execution output. It prompts for 8 elements (1 through 8) and displays the final array: 1 2 3 4 5 6.
- Taskbar:** At the bottom, it shows the Windows taskbar with various application icons and system information like temperature (34°C) and date (08-03-2022).

Q3

Aim:

Write a 'C' program perform matrix addition for n x n matrix. Get the user input for 'n'.

Procedure:

Input:

Number of rows and columns, n

Matrix A and Matrix B with n*n elements

Output:

Matrix which is addition of Matrix A and Matrix B

Algorithm:

Step 1: Read n

Step 2: Initialise matrix A and matrix B of size n*n

Step 3: Use nested for loop to read elements in the matrix A and matrix B

Step 4: Use nested for loop to print elements of matrix A and matrix B

Step 5: Initialise addition matrix, MA of size n*n

Step 5: Use nested for loops with counter i and j

$$MA[i][j] = A[i][j] + B[i][j]$$

Step 6: Display addition matrix using nested for loops

Code:

```
#include <stdio.h>

int main() {

    // Initialise variables and Read n
    int n, i, j;
    printf("Number of rows and columns for matrix: ");
    scanf("%d",&n);

    // Read and display elements of matrix A and B
    int a[n][n], b[n][n];

    printf("Enter value for matrix A:\n");
    for (i = 0; i < n; i++) {
        for (j = 0; j < n; j++) {
            scanf("%d", &a[i][j]);
        }
        printf("\n");
    }

    printf("Matrix A:\n");
    for (i = 0; i < n; i++) {
        for (j = 0; j < n; j++) {
```

```

        printf("%d ", a[i][j]);
    }
    printf("\n");
}

printf("\nEnter value for matrix B: \n");
for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++) {
        scanf("%d", &b[i][j]);
    }
    printf("\n");
}

printf("Matrix B:\n");
for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++) {
        printf("%d ", b[i][j]);
    }
    printf("\n");
}

//Matrix Addition
int ma[n][n];
for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++) {
        ma[i][j] = a[i][j] + b[i][j];
    }
}

//Display Matrix Addition
printf("\nMatrix Addition of Matrix A and Matrix B:\n");
for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++) {
        printf("%d ", ma[i][j]);
    }
    printf("\n");
}
}

```

```

1 #include <stdio.h>
2
3 int main() {
4     // Initialize variables and Read n
5     int n1, n2, j;
6     printf("Number of rows and columns for matrix: ");
7     scanf("%d %d", &n1, &n2);
8
9     // Read and display elements of matrix A and B
10    int a[n1][n2], b[n1][n2];
11
12    printf("Enter value for matrix A:\n");
13    for (i = 0; i < n1; i++) {
14        for (j = 0; j < n2; j++) {
15            scanf("%d", &a[i][j]);
16        }
17        printf("\n");
18    }
19
20    printf("Matrix A:\n");
21    for (i = 0; i < n1; i++) {
22        for (j = 0; j < n2; j++) {
23            printf("%d ", a[i][j]);
24        }
25        printf("\n");
26    }
27
28    printf("Enter value for matrix B: \n");
29    for (i = 0; i < n1; i++) {
30        for (j = 0; j < n2; j++) {
31            scanf("%d", &b[i][j]);
32        }
33        printf("\n");
34    }
35
36    printf("Matrix B:\n");
37    for (i = 0; i < n1; i++) {
38        for (j = 0; j < n2; j++) {
39            printf("%d ", b[i][j]);
40        }
41        printf("\n");
42    }
43
44    //Matrix Addition
45    int m[n1][n2];
46    for (i = 0; i < n1; i++) {
47        for (j = 0; j < n2; j++) {
48            m[i][j] = a[i][j] + b[i][j];
49        }
50    }
51
52    //Display Matrix Addition
53    printf("Matrix Addition of Matrix A and Matrix B:\n");
54    for (i = 0; i < n1; i++) {
55        for (j = 0; j < n2; j++) {
56            printf("%d ", m[i][j]);
57        }
58        printf("\n");
59    }
60 }

```

```

PS C:\Users\Vidhi Shah\Desktop\VTTC Sem 2\C++ Lab\PP5> gcc -o matrixaddition matrixaddition.c
PS C:\Users\Vidhi Shah\Desktop\VTTC Sem 2\C++ Lab\PP5> ./matrixaddition
Number of rows and columns for matrix: 3
Enter value for matrix A:
1
2
3
4
5
6
7
8
9
Matrix A:
1 2 3
4 5 6
7 8 9
Enter value for matrix B:
10
11
12
13
14
15
16
17
18
Matrix B:
10 11 12
13 14 15
16 17 18
Matrix Addition of Matrix A and Matrix B:
11 13 15
17 19 21
23 25 27
PS C:\Users\Vidhi Shah\Desktop\VTTC Sem 2\C++ Lab\PP5>

```

33°C Mostly sunny 19:40 08-03-2022