

```
//q1)Search an element in a 1D Array using binary search.
#include <stdio.h>
int main()
{
    int i, arr[100], search, first, last, middle;
    for (i = 0; i < 10; i++)
    {
        printf("Enter %d No element (in ascending order):",i);
        scanf("%d", &arr[i]);
    }
    printf("\nEnter element to be searched");
    scanf("%d", &search);
    first = 0;
    last = 9;
    middle = (first + last) / 2;

    while (first <= last)
    {
        if (arr[middle] < search)
            first = middle + 1;
        else if (arr[middle] == search)
        {
            printf("\nThe number,%d found at Position %d", search, middle + 1);
            break;
        }
        else
            last = middle - 1;
        middle = (first + last) / 2;
    }
    if (first > last)
        printf("\nThe number, %d found at Position %d", search, middle + 1);
    return 0;
}
```

Output-

```
Enter 0 No element (in ascending order):2
Enter 1 No element (in ascending order):4
Enter 2 No element (in ascending order):5
Enter 3 No element (in ascending order):6
Enter 4 No element (in ascending order):7
Enter 5 No element (in ascending order):12
Enter 6 No element (in ascending order):16
Enter 7 No element (in ascending order):23
Enter 8 No element (in ascending order):25
Enter 9 No element (in ascending order):31
```

Enter element to be searched7

The number,7 found at Position 5

//q2)Sort the elements of a 1D array using selection sort

```
#define SIZE 10 #include<stdio.h> int
main(){ int arr[SIZE]; int i,j,temp;
printf("Enter elements of the array: \n");
for(i=0;i<SIZE;i++){ scanf("%d",&arr[i]);
}
for(i=0;i<SIZE-
1;i++){ for(j=i+1;j<SIZE;j++){

if(arr[i]>arr[j]){ temp=ar
r[i]; arr[i]=arr[j];
arr[j]=temp;
}
}
}
printf("The Sorted Array is:\n");
for(i=0;i<SIZE;i++){ printf("%d\
t", arr[i]);
}
printf("\n");
```

```
}
```

Output

Enter elements of the array:

23

43

56

24

87

342

23

77

21

11

The Sorted Array is:

11 21 23 23 24 43 56 77 87 342

//q3)Add Two Matrix

```
#define ROW 4
```

```
#define COL 3 #include
```

```
<stdio.h>
```

```
int main() {
```

```
int i, j;
```

```
int mat1[4][3] = {{11, 12, 13}, {14, 15, 16}, {17, 18, 19}, {20, 21, 22}};
```

```
int mat2[4][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}, {4, 9, 1}};  
int mat3[4][3];
```

```
for (i = 0; i < ROW; i++)  
{    for (j = 0; j < COL; j++) {  
        mat3[i][j] = mat1[i][j] + mat2[i][j];  
    }  
}  
  
printf("The resultant matrix mat3 is: \n");  
for (i = 0; i < ROW; i++)  
{    for (j = 0; j < COL; j++) {  
        printf("%d ", mat3[i][j]);  
    }  
    printf("\n");  
}  
return 0;  
}
```

Output-

The resultant matrix mat3 is:

12 14 16

18 20 22

24 26 28

24 30 23

//q4

```
#include<stdio.h>  
#define ROW1 3  
#define COL1 4  
#define ROW2 COL1  
#define COL2 3
```

```
int main() {
    int mat1[ROW1][COL1], mat2[ROW2][COL2], mat3[ROW1][COL2];
    int i, j, k;

    printf("Enter the matrix mat1(%d x %d) row-wise:\n", ROW1, COL1);
    for(i = 0; i < ROW1; i++)    for(j = 0; j < COL1; j++)
        scanf("%d", &mat1[i][j]);

    printf("Enter the matrix mat2(%d x %d) row-wise:\n", ROW2, COL2);
    for(i = 0; i < ROW2; i++)    for(j = 0; j < COL2; j++)
        scanf("%d", &mat2[i][j]);

    for(i = 0; i < ROW1; i++)
    {
        for(j = 0; j < COL2; j++)
        {
            mat3[i][j] = 0;
            for(k = 0; k < COL1; k++) {
                mat3[i][j] += mat1[i][k] * mat2[k][j];
            }
        }
    }

    printf("The resultant matrix is:\n");
    for(i = 0; i < ROW1; i++) {
        for(j = 0; j < COL2; j++) {
            printf("%5d", mat3[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

Output-

Enter the matrix mat1(3 x 4) row-wise:

1 2 3

4 5 6

7 8 1

9 12 2

Enter the matrix mat2(4 x 3) row-wise:

1 2 3 4

6 7 8 9

1 2 10 5

The resultant matrix is:

41 81 40

101 189 104

137 184 88

```
//q5)Find Transpose of a matrix
#define ROW 4
#define COL 3
#include <stdio.h>
int main() { int
mat1[ROW][COL],mat2[COL][ROW],i,j;
printf("Enter elements of matrix :\n");
for(i=0;i<ROW;i++) for(j=0;j<COL;j++)
scanf("%d", &mat1[i][j]);
for(i=0;i<COL;i++) for(j=0;j<ROW;j++)
mat2[i][j]=mat1[j][i]; printf("The
transpose of matrix is:\n");
for(i=0;i<COL;i++){ for(j=0;j<ROW;j++){
printf("%5d",mat2[i][j]);
}
printf("\n");
}
}
```

Ouput-

```
Enter elements of matrix :
1 2 3
4 5 6
7 8 9
1 10 3
The transpose of matrix is:
1 4 7 1
2 5 8 10
3 6 9 3
```

H.W QUESTIONS

```
//p1)
```

```
#include <stdio.h>

int main() {    float curr_year[12] = {3.2, 2.2, 3.6, 4.0, 4.0, 1.6, 2.4, 3.5, 4.2,
5.1, 3.7, 2.8};    float prev_year[12] = {4.0, 3.0, 3.2, 3.5, 3.8, 2.2, 2.6, 3.8, 4.0,
5.0, 4.0, 3.5};

    // Compute the total rainfall for each year and the average monthly rainfall
    float curr_year_total = 0, prev_year_total = 0;
    float curr_year_avg = 0, prev_year_avg = 0;
    int i;
    for (i = 0; i < 12; i++)
    {
        curr_year_total += curr_year[i];
        prev_year_total += prev_year[i];
    }
    curr_year_avg = curr_year_total / 12;
    prev_year_avg = prev_year_total / 12;

    // Print the table of monthly rainfall and comparison with the previous year
    printf("Table of monthly rainfall\n");
    printf("%-10s%-10s%-10s\n", "", "This year", "Last year");    char* month_names[12]
= {"January", "February", "March", "April", "May", "June", "July",
"August", "September", "October", "November", "December"};    for (i = 0; i < 12; i++)
    {
        printf("%-10s%-10.1f%-10.1f\n", month_names[i], curr_year[i], prev_year[i]);
    }
    printf("Total rainfall this year: %.1f\n", curr_year_total);
    printf("Total rainfall last year: %.1f\n", prev_year_total);
    printf("Average monthly rainfall for this year: %.1f\n", curr_year_avg);
    printf("Average monthly rainfall for last year: %.1f\n", prev_year_avg);

    return 0;
}
```

Ouput-

Table of monthly rainfall

	This year	Last year
January	3.2	4.0
February	2.2	3.0
March	3.6	3.2
April	4.0	3.5
May	4.0	3.8
June	1.6	2.2
July	2.4	2.6
August	3.5	3.8
September	4.2	4.0
October	5.1	5.0
November	3.7	4.0
December	2.8	3.5
Total rainfall this year: 40.3		

Total rainfall last year: 42.6
Average monthly rainfall for this year: 3.4
Average monthly rainfall for last year: 3.5

```
//p2
#include <stdio.h>

int main() {
    // Define the data for the candidates and precincts
    int votes[5][4] = {{350, 200, 180, 205}, {260, 225, 185, 200}, {300, 180, 215, 205}, {225, 225, 220, 108}, {175, 250, 195, 230}};
    char candidates[5] = {'A', 'B', 'C', 'D', 'E'};
    char precincts[4][10] = {"Precinct 1", "Precinct 2", "Precinct 3", "Precinct 4"};
    int total_votes[5] = {0};
    float percent_votes[5] = {0};

    // Display the table of votes
    printf("Table of Votes\n");
    printf("Candidate precincts[0] precincts[1] precincts[2] precincts[3]\n");
    for (int i = 0; i < 5; i++) {    printf("%5c  ", candidates[i]);    for (int j
= 0; j < 4; j++) {        printf("%10d ", votes[i][j]);
        total_votes[i] += votes[i][j];
    }
    percent_votes[i] = ((float)total_votes[i] / 980) * 100;
    printf("%10d\n", total_votes[i]);
}
printf("\n");

    // Compute and display the total votes and percentage of votes for each candidate
    printf("Total Votes and Percentage of Votes\n");    for (int i = 0; i < 5; i++) {
        printf("%5c  %10d  %.2f%%\n", candidates[i], total_votes[i], percent_votes[i]);
    }
    printf("\n");

    // Check if any candidate received over 50% of the votes
    int winner = -1;    for (int i = 0; i < 5; i++) {        if
(percent_votes[i] > 50.0) {
        winner = i;
        printf("Winner: Candidate %c\n", candidates[winner]);
        break;
    }
}

    // If no candidate received over 50% of the votes, identify the two candidates with the highest
votes and declare a runoff    if (winner == -1) {        int max1 = -1, max2 = -1;
        for (int i = 0; i < 5; i++)
        {            if (total_votes[i] > max1)
```

```
{
    max2 = max1;
max1 = total_votes[i];
    }
    else if (total_votes[i] > max2) {
        max2 = total_votes[i];
    }
    }
    printf("Runoff between Candidates %c and %c\n", candidates[max1], candidates[max2]);
}

return 0;
}
```

Output-

Table of Votes

Candidate precincts[0] precincts[1] precincts[2] precincts[3]

A	350	200	180	205	935
B	260	225	185	200	870
C	300	180	215	205	900
D	225	225	220	108	778
E	175	250	195	230	850

Total Votes and Percentage of Votes

A	935	95.41%
B	870	88.78%
C	900	91.84%
D	778	79.39%
E	850	86.73%

Winner: Candidate A

//p3

#include <stdio.h>

```
int main() {
    int i, j;
    int black_women[6][2] = {{68, 74}, {70, 76}, {72, 78}, {74, 80}, {76, 82}, {78, 84}};
    int black_men[6][2] = {{63, 68}, {65, 70}, {67, 72}, {69, 74}, {71, 76}, {73, 78}};    int
    white_women[6][2] = {{73, 79}, {75, 81}, {77, 83}, {79, 85}, {81, 87}, {83, 89}};    int
    white_men[6][2] = {{67, 73}, {69, 75}, {71, 77}, {73, 79}, {75, 81}, {77, 83}};    int
    diff[6][2];

    printf("Life expectancy matrix for black women and men:\n");
    for (i = 0; i < 6; i++) {
        printf("%d\t%d\n", black_women[i][0], black_men[i][0]);
    }
}
```



```
printf("\nLife expectancy matrix for white women and men:\n");
for (i = 0; i < 6; i++) {
    printf("%d\t%d\n", white_women[i][0], white_men[i][0]);
}

printf("\nDifference matrix:\n");
for (i = 0; i < 6; i++) {
    for (j = 0; j < 2; j++) {
        diff[i][j] = black_women[i][j] - black_men[i][j];
        printf("%d\t", diff[i][j]);
    }
    printf("\n");
}

return 0;
}
```

Output-

Life expectancy matrix for black women and men:

68	63
70	65
72	67
74	69
76	71
78	73

Life expectancy matrix for white women and men:

73	67
75	69
77	71
79	73
81	75
83	77

Difference matrix:

5	6
5	6
5	6
5	6
5	6
5	6