Arrays and Subscripted variables

A list of related values is stored in RAM in the form of an array. Values in an array are identified using array name with subscripts. Single subscripted variable is referred to a one-dimensional or linear array; two subscripted variable is referred to a two-dimensional or matrix array and so on.

The syntax of array in C.

An array is declared with specific number of elements in type declaration statement. It has the following form.

type name[n];

where type is the data type of values in the array.
name is the array name.
n is the maximum number of values that can be stored in the array.
Example: int x[100], a[3][4];
This statement declares a one–dimensional array with 100 elements and a two-dimensional array with 12 elements (3 X 4 = 12).

Q. What are the characteristics of arrays in C?

Solution: The characteristics of arrays in C are:
1) An array holds elements that have the same data type
2) Array elements are stored in subsequent memory locations
3) Two-dimensional array elements are stored row by row in subsequent memory locations.
4) Array name represents the address of the starting element
5) Array size should be mentioned in the declaration. Array size must be a constant expression and not a variable.

Q. What is an array? Write down the syntax to declare a one dimensional array and a two dimensional array.

Solution:

An **Array** is a collection of identical data elements which are stored in consecutive memory locations under a common heading or a variable name. The individual values in an array are called its elements.

The array is also defined same as other variables before it is used in the program. Defining the name and the type of an array and setting the number of elements in the array is known as dimensioning the array.

In general, the syntax to declare a one dimensional array is as:

data_type array_name[expression];

where, **data_type** refers to the nature of the data elements stored in array.

array_name is the name of the array.

expression is used to declare the size of the array for further processing.

For example, int x[20]; float y[10]; etc., are some one dimensional array.

In general, the syntax to declare a two dimensional array is as:

data_type array_name[expression 1][expression 2];

where, **data_type** refers to the nature of the data elements stored in array.

array_name is the name of the array.

expression 1 indicates the number of rows.

expression 2 indicates the number of columns.

For example: int x[10]10]; float y[5][5]; etc., are some two dimensional array.

Q. What is an array? How array can be defined? Explain with syntax and example. How are array elements stored in memory?

Solution:

An **Array** is a collection of identical data elements which are stored in consecutive memory locations under a common heading or a variable name. The individual values in an array are called its elements.

The array is also defined same as other variables before it is used in the program. Defining the name and the type of an array and setting the number of elements in the array is known as dimensioning the array.

In general, a one dimensional array may be expressed as:

data_type array_name[expression];

where, **data_type** refers to the nature of the data elements stored in array.

array_name is the name of the array.

expression is used to declare the size of the array for further processing.

Examples of one dimensional array declaration are:

```
int a[100];    /*an integer type array of size 100 */
char name[50];      /* a char type array of size 50 */
```

The general format of one dimensional array initialization is:

data_type array_name[expression]={element1, element2, …. Element n};

Example:

```
int a[4] = {10,11,12,13};
char sex[2] = {'m','f'};
int queue[4] = {1,2};
```

In C, arrays are zero-based: the ten elements of a 10-element array are numbered from 0 to 9. The subscript which specifies a single element of an array is simply an integer expression in square brackets. The first element of the array b is b[0], the second element is b[1], etc.

The above examples will be stored in the memory as:

a[0]=10	sex[0]='m'	queue[0]=1
a[1]=11	sex[1]='f'	queue[1]=1
a[2]=12		queue[2]=0
a[3]=13		queue[3]=0

Note: In third example the array is of integer type with maximum length of four elements, but only two elements are initialized therefore the next two elements are taken as of value 0 in default.

Q. How do you initialize arrays in C?

Solution: Arrays are initialized while they are declared in type declaration statement. Consider the following example:

```
int x[5] = {5, 25, 32, -30, 22};
```

This statement initializes x[0] = 5, x[1] = 25, x[2] = 32, x[3] = -30, and x[4] =22.

```
int a[3][3] = {{1, 0, 0}, {0, 1, 0}, {0, 0, 1}};
```

This statement initializes a unit matrix of order 3 x 3. Note that each row of the matrix is enclosed in { }.

Q. What is an array variable? How does it differ from an ordinary variable?

Solution: An array variable or a subscripted variable is written along with array name followed by one or more subscript in brackets. An array variable is declared in type declaration statement along with the size of the array.

An ordinary variable or scalar variable is written without any subscript, and is always referring to a single value.

Q. How is an array of 10 integers declared and used in C?

Solution: An array of 10 integers is declared as follows:

```
int x[10];
```

The values of the array are accessed using the array name with one subscript in bracket. For example, x[0], x[1], …., x[9].

Normally array values are read/written using a for loop. For example,

```
for(i = 0; i<= 10; i++)
        scanf("%d", &x[i]);
```

Q. Describe the array defined in the following statement. Indicate what values are assigned to the individual array elements.

```
int p[2][4] = {{1,3},{5,7}};
```

Solution:

The statement:

```
int p[2][4]={{1,3},{5,7}};
```

mean, p is a two dimensional (2 X 4) array and is initialized elements of the array.

The values are assigned to the individual elements is given below:

p[0][0]=1
p[0][1]=3
p[0][2]=0
p[0][3]=0

p[1][0]=5
p[1][1]=7
p[1][2]=0
p[1][3]=0

Note: Here, only four elements are initialized therefore the next two elements of the first row are taken as of value 0 by default. Similarly , the next two elements of the second row are taken as of value 0 by default.

Q. What is the purpose of a looping statements in a programming language? What is the advantage of using an array in a programming language? Write four expression using an array.

Solution: Looping statements are provided in a programming language to support execution of statement(s) repeatedly.

Array provides a very simple and efficient way of referring to and performing computation on collection of data that have got some common attributes.

Four expressions are:

a[3]=a[2] + a[i];
a[i]=5;
a[2]=5*a[i];
a[i]=a[i]+6;

Q .32. Write a program to multiply two matrices and print the result in a matrix form.

Solution:

```
/* Multiply two matrices */
#include<stdio.h>
#define S 5
main()
{
   int a[S][S], b[S][S], c[S][S],i,j,m,n,p,q;
   void enter();
   void display();
   void product();
   printf("Enter the order of first matrix <=%d * %d : ",S,S);
   scanf("%d%d",&m,&n);
   printf("%d%5d\n",m,n);
   printf("Enter the order of second matrix<=%d * %d : ",S,S);
   scanf("%d%d",&p,&q);
```

```
    printf("%d%5d\n",p,q);
    if(n= =p)
    {
       printf("\nEnter the first matrix of order %d * %d\n",m,n);
       enter(a,m,n);
       display(a,m,n);
       printf("\nEnter the second matrix of order %d * %d\n",p,q);
       enter(b,p,q);
       display(b,p,q);
       product(a,b,c,m,n,q);
       printf("\nResultant matrix is:\n");
       display(c,m,q);
    }
    else
       printf("\nMatrix multiplication not possible\n");
} /* End of main() */

void enter(int x[][S], int row, int colm)
{
  int i,j;
  for(i=0;i<row;i++)
     for(j=0;j<colm;j++)
        scanf("%d",&x[i][j]);
}

void display(int x[][S], int row, int colm)
{
  int i,j;
  for(i=0;i<row;i++)
  {
     for(j=0;j<colm;j++)
        printf("%5d",x[i][j]);
     printf("\n");
  }
}

void product(int x[][S], int y[][S], int z[][S], int row1, int colm1, int
colm2)
{
   int i,j,k;
   for(i=0;i<row1;i++)
   {
```

```
    for(j=0;j<colm2;j++)
    {
      z[i][j]=0;
      for(k=0;k<colm1;k++)
        z[i][j]+=x[i][k]*y[k][j];
    }
  }
}
```

Run1:

```
Enter the order of first matrix <=5 * 5 : 3    3
Enter the order of second matrix<=5 * 5 : 3    3

Enter the first matrix of order 3 * 3
   1    4    2
   6    0    5
   7    8    3

Enter the second matrix of order 3 * 3
   9    0    1
   3    5    2
   4    1    8

Resultant matrix is:
  29   22   25
  74    5   46
  99   43   47
```

Run2:

```
Enter the order of first matrix <=5 * 5 : 3    2
Enter the order of second matrix<=5 * 5 : 3    4

Matrix multiplication not possible
```

Q. Write a program to transpose a matrix (the transpose can be obtained by interchanging the elements of rows and columns).

Solution:

```
/* Program to transpose a given matrix */
#include<stdio.h>
main( )
{
```

```
    int mat1[20][20],mat2[20][20],i,j,m,n;
    printf("Give the number of row & column of matrix:" );
    scanf("%d %d",&m,&n);
    printf("%d %3d\n",m,n);
    printf("Now input  %d elements of matrix:\n",m*n);
    for(i=1;i<m+1;i++)
    {
        for(j=1;j<n+1;j++)
            scanf("%d",&mat1[i][j]);
    }
    printf("\n\t original matrix: \n");
    for(i=1;i<m+1;i++)
    {
        for(j=1;j<n+1;j++)
            printf("%4d",mat1[i][j]);
        printf("\n");
    }
    for(i=1; i<n+1;i++)
    {
        for(j=1;j<m+1;j++)
            mat2[i][j]=mat1[j][i];
    }
    printf("\n\t Transpose of matrix: \n");
    for(i=1;i<n+1;i++)
    {
        for(j=1;j<m+1;j++)
            printf("%4d",mat2[i][j]);
    }
    printf("\n");
}
```

Run:

```
Give the number of  row and column of matrix:  3  3
Now input 9 elements of matrix
1 2 3 4 5 6 7 8 9
Original matrix
   1   2   3
   4   5   6
   7   8   9
Transpose of matrix:
   1   4   7
   2   5   8
```

3 6 9

Q.28. Write a program to add and subtract the elements of two matrices.

Solution:

```
/* Program to add and subtract the elements of two matrices */
#include<stdio.h>
main( )
{
        int a[10][10],b[10][10],c[10][10],d[10][10];
        int i,j, m, n, p, q;
        printf("Give the number of rows & columns of A matrix\n");
        scanf("%d %d", &n,&m);
        printf("%d %d\n",n,m);
        printf("Give the number of rows & columns of B matrix\n");
        scanf("%d %d",&p,&q);
        printf("%d %d\n",p,q);
        /* Check if matrices can be added */
        if(n = = p && m = = q)
        {
                printf("Matrices can be added\n");
                printf("Now input elements of matrix A\n");
                for(i=0;i<n;++i)
                        for(j=0;j<m;++j)
                                scanf("%d",&a[i][j]);
                /* Print – A matrix */
                printf("The matrix A is:\n");
                for(i=0;i<n;i++)
                {
                        for(j=0;j<m;++j)
                        printf("%5d",a[i][j]);
                        printf("\n");
                }
                printf("Now input elements of matrix B\n");
                for(i=0;i<n;++i)
                        for(j=0;j<m;j++)
                                scanf("%d",&b[i][j]);
                /* Print – B matrix */
                printf("The matrix b is:\n");
                for(i=0;i<n;++i)
                {
                        for(j=0;j<m;++j)
```

```
                printf("%5d",b[i][j]);
            printf("\n");
        }
        /* Addition of  elements of two matrices */
        for(i=0;i<n;++i)
        {
            for(j=0;j<m;++j)
                c[i][j]=a[i][j] + b[i][j];
        }
        printf("Addition of elements of A and B matrices :\n");
        for(i=0;i<n;++i)
        {
            for(j=0;j<m;++j)
                printf("%5d",c[i][j]);
            printf("\n");
        }
        /* Subtraction of elements of B matrix from A */
        for(i=0;i<m;++i)
        {
            for(j=0;j<m;++j)
                d[i][j] = a[i][j] – b[i][j];
        }
        printf("Subtraction of elements of matrix B from A\n");
        for(i=0;i<n;++i)
        {
            for(j=0;j<m;++j)
            printf("%5d",d[i][j]);
            printf("\n");
        }
        }
        else
        {
            printf("Matrices cannot be added or subtracted\n");
}
}
```

Run:

Give the number of rows & columns of A matrix 2 2
Give the number of rows & columns of B matrix 2 2
Matrices can be added
Now input elements of matrix A

```
9 8 7 5
The matrix A is:
   9   8
   7   5
Now input elements of matrix B
2 4 3 4
The matrix B is:
   2   4
   3   4
Addition of the elements of A and B matrices:
   11 12
   10 9
Subtraction of the elements of matrix B from A:
    7    4
    4    1
```

Q. Write a program to read the values for the elements of a one-dimensional array of integers and then sorts it in the ascending order, i.e. minimum to maximum.

OR

Write a program to sort a set of n numbers in ascending order.

Solution:

```
/* Sorts an array in ascending order*/
main( )
{
        int a[100];
        int i, j, n, temp, evensum;
        evensum = 0;
        printf("How many numbers are in the array?  ");
        scanf("%d",&n);
        printf("%d\n",n);
        printf("Enter the elements: \n");
        for(i=0;i<=n-1;++i)
        {
                scanf("%d",&a[i]);
        }
        printf("Contents of the array(unsorted form\n");
```

```
        for(i=0;i<=n-1;++i)
                printf("%d\t",a[i]);
        printf("\n");
        for(i=0;i<=n-1;++i)
        {
                for(j=0;j<=n-1;++j)
                if(a[i]<a[j])
                {
                        temp =a[i];
                        a[i]=a[j];
                        a[j]=temp;
                }
        }
        printf("Contents of the array(sorted form)\n");
        for(i=0;i<=n-1;++i)
                printf("%d\t",a[i]);
}
```

Q. Write a program in C to find the maximum and minimum of an array of numbers.

Solution: The program in C to find the maximum and minimum of an array of numbers is shown below:

```
/* Program to find maximum & minimum number in an array*/
#include<stdio.h>
#include<conio.h>
main( )
{
        int n,i, j, num[10], max, min,temp;
        clrscr();
        printf("How many numbers in the array?  ");
        scanf("%d",&n);
        printf("%d\n",n);
        printf("Enter %d numbers:\n",n);
        for(i=0;i<n;i++)
        {
                scanf("%d",&num[i]);
        }
        for(i=0;i<n;i++)
                printf("%5d",num[i]);
        printf("\n");
        max=num[0];
```

```
        min=num[0];
        for(i=0;i<n;i++)
        {
                if(num[i]>max)
                        max=num[i];
                if(num[i]<min)
                        min=num[i];
         }
        printf("The maximum number is %d\n",max);
        printf("The minimum number is %d\n",min);
        printf("\nPress any key to exit...\n");
        getch();
        return;
}
```

The output of the above program will be:

```
How many numbers in the array?  8
Enter 8 numbers:
   4   87    2   -5   89    0    2    1
The maximum number is 89
The minimum number is -5

Press any key to exit...
```

Q. Write a program in C to add two 4 x 4 matrix and print the diagonal elements of the resultant matrix.

Solution:

```
/* To add two matrices & print the diagonal elements of the resultant
matrix */
#include<stdio.h>
#include<conio.h>
main( )
{
        int a[10][10],b[10][10],c[10][10],d[10][10];
        int i,j;
        clrscr();
        printf("Input 16 elements of matrix A\n");
        for(i=0;i<4;++i)
        for(j=0;j<4;++j)
                scanf("%d",&a[i][j]);
        /* Print - A matrix */
```

```
        for(i=0;i<4;i++)
        {
                for(j=0;j<4;++j)
                        printf("%5d",a[i][j]);
                printf("\n");
        }
        printf("Input 16 elements of matrix B\n");
        for(i=0;i<4;++i)
                for(j=0;j<4;j++)
                        scanf("%d",&b[i][j]);
        /* Print - B matrix */

        for(i=0;i<4;++i)
        {
                for(j=0;j<4;++j)
                        printf("%5d",b[i][j]);
                printf("\n");
        }
        /* Addition of  elements of two matrices */
        for(i=0;i<4;++i)
        {
                for(j=0;j<4;++j)
                        c[i][j]=a[i][j] + b[i][j];
        }
        printf("Addition of elements of A and B matrices :\n");
        for(i=0;i<4;++i)
        {
                for(j=0;j<4;++j)
                        printf("%5d",c[i][j]);
                printf("\n");
        }
        printf("The diagonal elements of the resultant matrix is:\n");
        for(i=0;i<4;i++)
                printf("%4d",c[i][i]);


        printf("\nPress any key to exit...\n");
        getch();
        return;
}
```

The output of the above program will be:

Input 16 elements of matrix A

```
   1   2   3   4
   5   6   7   8
   9   8   7   6
   5   4   3   2
Input 16 elements of matrix B
   1   1   1   1
   2   2   2   3
   4   5   6   7
   8   9   3   4
Addition of elements of A and B matrices :
   2   3   4   5
   7   8   9  11
  13  13  13  13
  13  13   6   6
The diagonal elements of the resultant matrix is:
  2   8  13   6
Press any key to exit...
```

Q. Write a program to read n real numbers and print the standard deviation of the numbers where the standard deviation of n numbers x_1, x_2, …, x_n is given by

$$SD = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n}} \qquad \text{where,} \quad \bar{x} = \sum_{i=1}^{n} x_i / n$$

Solution:

```
/* Calculating mean, variance and standard deviation */
#include<stdio.h>
#include<math.h>
main( )
{
        int i, n;
        float x[50], mean, variance, sdn;
        float sumsq=0.0, sum=0.0;
        printf("Calculating standard deviation of a list of items.\n");
        printf("\nEnter size of the list:");
        scanf("%d",&n);
        printf("%d\n", n);
        printf("\nEnter the %d items\n",n);
```

```
        for(i=0;i<n; i++)
        {
                scanf("%f", &x[i]);
                sum+=x[i];
        }
        for(i=0; i<n; i++)
        printf("%8.2f", x[i]);
        mean = sum/(float)n;
        /* Computing variance */
        for(i=0; i<n; i++)
                sumsq=sumsq + (mean – x[i]*(mean – x[i]));
        variance= sumsq/(float)n;
        /* Standard deviation */
        sdn = sqrt(variance);
        printf("\n\nMean of %5d items: %10.6f\n",n,mean);
        printf("\nVariance :%10.6f\n", variance);
        printf("\nStandard Deviation: %10.6f\n", sdn);
}
```

The output of the above program will be:

```
Calculating standard deviation of a list of items
Enter size of the list: 7
Enter the 7 items
32.00 11.00 90.00 34.00 52.00 24.00 7.00
Mean of 7 items: 35.714287
Variance: 685.918396
Standard Deviation: 26.190044
```

Q.31: What do you mean by square matrix? Write a C program to find the sum of diagonal elements of such a matrix.

Solution: A matrix is called square matrix, if the number of rows of the matrix is equal to the number of column.

The program to find the sum of diagonal elements of a square matrix is given below:

```
/* Program to find the sum of diagonal elements of a square matrix */
#include<stdio.h>
#include<conio.h>
main( )
{
        int a[10][10],sum=0;
        void read(int x[][10],int);
```

```
        void display(int x[10][10], int);
        int i,j, n;
        clrscr();
        printf("Give the size of the matrix\n");
        scanf("%d", &n);
        printf("%d  %d\n",n,n);
        printf("Input the elements of matrix:\n");
        read(a,n);
        printf("The matrix A is:\n");
        display(a,n);
        /* Sum of the diagonal elements */
        for(i=0;i<n;++i)
        {
                sum+=a[i][i];
        }
        printf("Sum of the diagonal elements is %d\n", sum);
        printf("\nPress any key to exit...\n");
        getch();
        return;
} /* End of main() */

void read(int x[][10], int r)
{
        int i,j;
  for(i=0;i<r;i++)
        for(j=0;j<r;j++)
         scanf("%d",&x[i][j]);
}
void display(int x[][10], int r)
{
  int i,j;
  for(i=0;i<r;i++)
  {
    for(j=0;j<r;j++)
       printf("%5d",x[i][j]);
    printf("\n");
  }
}
```

The output of the above program will be:

```
Give the size of the matrix
3  3
Input the elements of matrix:
```

```
The matrix A is:
   1   2   3
   4   7   6
   9   3   4
Sum of the diagonal elements is 12

Press any key to exit...
```

Q. Write a program in C to find the sum of n given numbers, where n is also given as input.

Solution:

```
/* Program to find the sum of n given numbers */
#include<stdio.h>
#include<conio.h>
main( )
{
        int a[10],sum=0,i,n;
        clrscr();
        printf("How many number? ");
        scanf("%d",&n);
        printf("%d\n",n);
        printf("Enter %d numbers:\n",n);
        for(i=0;i<n;i++)
                scanf("%d",&a[i]);
        for(i=0;i<n;i++)
                printf("%4d",a[i]);
        printf("\n");
        for(i=0;i<n;i++)
                sum+=a[i];
        printf("The sum of the numbers is=%d\n",sum);
        printf("\nPress any key to exit...\n");
        getch();
        return;
}
```

The output of the above program will be:

```
How many number? 6
Enter 6 numbers:
  3   1   6   3   9   2
The sum of the numbers is=24

Press any key to exit...
```

Q. Write a program to find the largest of 5 numbers.

Solution: The program to find the largest of 5 numbers is given below:

```
/* Program to find largest number among 5 numbers */
#include<stdio.h>
#include<conio.h>
main( )
{
        int x[5],i,larg;
        clrscr();
        printf("Enter 5 numbers:\n");
        for(i=0;i<5;i++)
                scanf("%d",&x[i]);
        for(i=0;i<5;i++)
                printf("%4d",x[i]);
        printf("\n");
        larg=x[0];
        for(i=0;i<5;i++)
          if(larg<x[i])
            larg=x[i];
        printf("The largest number is= %d\n",larg);
        printf("Press any key to exit...\n");
        getch();
        return 0;
}/* End of main( ) */
```

The output of the above program would be:

```
Enter 5 numbers:
  7   2   9  12   5
The largest number is= 12
Press any key to exit...
```

Q. Write a program to find the average of n given integers.

Solution: The program to find the average of n given integers is given below:

```
/* Program to find the average of the given numbers */
#include<stdio.h>
#include<conio.h>
main( )
```

```
{
        int x[20],i,n,sum=0;
        float avg;
        clrscr();
        printf("How many integers? ");
        scanf("%d",&n);
        printf("%d\n",n);
        printf("Enter %d numbers:\n",n);
        for(i=0;i<n;i++)
                scanf("%d",&x[i]);
        for(i=0;i<n;i++)
                printf("%4d",x[i]);
        printf("\n");
        for(i=0;i<n;i++)
          sum+=x[i];
        avg=(float)sum/n;
        printf("The average is= %.2f\n",avg);
        printf("Press any key to exit...\n");
        getch();
        return 0;
}/* End of main( ) */
```

The output of the above program will be:

```
How many integers? 5
Enter 5 numbers:
  4   6   3   7   8
The average is= 5.60
Press any key to exit...
```

Q. 1: Answer the following questions:

a. What is an array?
b. What is the subscript of the first element of an array in C?
c. How would you declare an array x containing 50 integer elements followed immediately by 50 real elements?
d. What purpose is served by the break statement?

e. When a one-dimensional array is being declared, under what condition may the dimension be omitted, with the array name followed by an empty pair of square brackets?
f. What is the maximum number of dimensions an array in C may have?
g. What happens if an array is being initialized within its declaration, and too few initialization values are specified within the curly braces?
h. What happens if the number of initializing values is greater than the dimension specified for the array?
i. Is it possible to declare and initialize an array in C simultaneously? If so, how?
j. How is the integer array a, containing 100 elements, declared in C?

Solution:

a. An array is an ordered collection of elements that share the same name.
b. 0 (Zero)
c. It can't be done. (All elements of a single array must be of the same type.).
d. It provides a means of immediately terminating the execution of a loop.
e. If the entire array is being initialized within the declaration.
f. Theoretically, there is no limit. The only practical limits are memory size and the restrictions imposed by the compiler being used.
g. This is perfectly acceptable, as long as the dimension is specified explicitly in the declaration. The initialization values that are specified are assigned to consecutive elements of the array, starting with element 0. The remaining elements are initialized to zero.
h. A syntax error occurs during compilation of the program, and the compilation is aborted.
i. Yes. The array declaration is followed immediately by an equal sign. This is followed by the list of values to be assigned (separated by commas) enclosed in braces.
j. int a[100];

&&&&&&&&